Selective Image Segmentation Models and Fast Multigrid Methods

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Michael T. Roberts

January 2019

Contents

Acknowledgements

I have an almost uncountable number of people to thank who have been a critical part of getting me to this stage of life and have made this thesis possible.

Firstly, I thank my parents for their support through my life and encouraging me to never limit the scope of my ambition. I also thank my wider family and in particular my grandmother for her relentless support of my wildest ideas.

Secondly, thanks are due to the mathematics teachers at Altrincham Grammar School for their encouragement to pursue mathematics to a higher level, even when I doubted my own capabilities.

Thirdly, I must thank my MMath supervisor Dr. Pavel Tumarkin for giving me a passion for research which ultimately pushed me to apply for this Ph.D.

Fourth, to my supervisor Prof. Ke Chen for his support of my research and facilitating many new opportunities for me which would never have been possible without him. Thanks are also due to Prof. Klaus Irion for giving me great insight and enthusiasm for studying lung disease and the particular application of my work to medical imaging. I also must thank Prof. Rachel Bearon and Prof. Carola-Bibiane Schönlieb who made my viva thoroughly enjoyable and suggested many improvements to this thesis.

Finally, I thank all the friends I have picked up along the way throughout my life. They have been long-suffering but shaped me into the person I am today. In particular, I must mention Aarti, Adam, Charlie, Gaby, Jonathan, Josh, Junaid, Kathryn, Kieran, Louis, Paul, Sam, Stephen, Thomas, Vinícius and Zac for particular thanks.

I am a Mancunian to my core, but Liverpool has left an indelible impression on my heart. I have thoroughly enjoyed my time here and have met friends I will have for a lifetime.

Abstract

This thesis is concerned with developing robust and accurate variational selective image segmentation models along with fast multigrid methods to solve non-linear partial differential equations (PDEs).

The first two major contributions are the development of new distance terms and new intensity fitting terms for selective image segmentation models. These give state-of-the-art segmentation results, with high robustness to the main parameters and to the user input. Therefore, these models are highly applicable to real-world applications such as segmenting single organs from medical scans.

The final major contribution is to develop new novel non-standard smoothers for the non-linear full approximation scheme multigrid framework. Multigrid is an optimal $\mathcal{O}(N)$ iterative scheme when it converges. However, typically if we directly apply a multigrid solver to a non-linear problem, it will not converge. This is principally due to the ineffectiveness of the standard smoothing schemes such as Jacobi or Gauss-Seidel. We review the true reason that these smoothers are ineffective using local Fourier analysis and develop a smoother which is guaranteed to be effective. Experiments show that the smoother is effective and the algorithm converges as desired. These new non-standard smoothing schemes can be used to solve a whole class of non-linear PDEs quickly. This work also lays the groundwork in the development of a "black box" non-linear multigrid solver which doesn't require the degree of tuning that current multigrid algorithms do.

Publications and Presentations

Publications

- M. Roberts, K. Chen, and K. L. Irion. A convex geodesic selective model for image segmentation. *To appear in Journal of Mathematical Imaging and Vision*, 2018.
- M. Roberts and J. Spencer. Chan-Vese reformulation for selective image segmentation. *To appear in Journal of Mathematical Imaging and Vision*, 2019.
- M. Roberts, K. Chen, and K. L. Irion. Multigrid algorithm based on hybrid smoothers for variational and selective segmentation models. *To appear in International Journal of Computer Mathematics*, 2018.
- M. Roberts, K. Chen, and K. L. Irion. On an effective multigrid solver for solving a class of variational problems with application to a unified image segmentation model. *Submitted*, 2018.
- Y. Guo, J. Li and M. Roberts. Shape reconstruction based on direct sampling and image segmentation. *In Preparation*.

Presentations

- On a New Multigrid Algorithm for Image Segmentation. SIAM Imaging Conference 2018. Bologna, Italy, June 2018.
- A New Robust Convex Selective Image Segmentation Model in 2D and 3D. Developments in Healthcare Imaging – Connecting with Academia. Cambridge, UK, May 2018.
- Fast and Robust Image Segmentation in 2D and 3D. The 27th Biennial Numerical Analysis Conference. Glasgow, UK, June 2017
- A Multigrid Solver for the Rada-Chen Selective Segmentation Model. SIAM Imaging Conference 2016. Albuquerque, USA, May 2016.

Chapter 1

Introduction

This thesis focusses on the overall theme of developing new, robust and applicable variational models for selective image segmentation and obtaining this solution quickly. Image segmentation is one of the most important applications of image processing techniques. The aim is to extract an object or objects from an image. There are two distinct types of image segmentation – global and selective.

Global image segmentation aims to segment all objects in an image and selective image segmentation aims to segment only the object (or objects) which we would like to segment. In this thesis we focus only on selective image segmentation as it has vast applicability and importance, particularly for medical imaging. Selective image segmentation generally requires some user input, for example the use of a marker set to indicate which object or objects in an image they desire to segment. Selective segmentation therefore permits the isolation of specific objects in an image, e.g. single or multiple organs in a medical scan, whereas global image segmentation would allows us to segment all objects in an image as one object – but not differentiate between them.

In this thesis we develop two new models for selective image segmentation.

The first model focusses on reformulating the notion of the distance between two pixels in an image. We move away from the Euclidean idea of distance to the Geodesic distance. We say that if the path between two pixels is of homogeneous intensity then the geodesic distance between them is small and, similarly, if there is an edge between two pixels then

the geodesic distance between them can be very large, even if the Euclidean distance is small.

The second model we develop builds upon the first. We find that for images which have similar average foreground and background intensities the current selective segmentation models can perform poorly. To solve this problem, we develop a new model which disregards the background intensity completely. This is rather intuitive, as with selective segmentation we should only have interest in the intensity of the object or objects we wish to segment. The background intensity is of little interest. This second model has vast clinical applications, as we show in Chapter 4, due to the fact that it is very robust to the precise location of the marker set and is also robust to changes in the two main tuning parameters. This allows a clinician to quickly draw a marker set within the object they wish to segment, only requiring that the whole marker set has three or more points selected and lies within the object in the image. It also means that with minimal parameter tuning we obtain accurate segmentation results, another benefit for a clinician who need not focus on parameter tuning to obtain the result they require.

The models we study are formulated in a variational framework. They are based on, typically non-linear, energy functionals which we aim to minimise. The variational approach is in contrast to the more "black-box" algorithms which simply give a result for an input image, an example being segmentation based on region growing using an initial seed set of marker point. The key benefit of the variational approach is that it has a rigorous mathematical basis, which allows us to obtain certain assurances on existence and uniqueness of the PDEs which result from the minimisation of the energy functionals. We also have notions of convexity and non-convexity for the energy functionals, allowing us to develop convex models which have only one minimum value (the global minimum) and hence we can achieve the same segmentation results regardless of the initial conditions used. Another benefit of the variational approach is that we have the ability to trade-off between certain properties that we wish the solution to possess, such as the smoothness of the segmentation boundary, the length of the segmentation boundary and ensuring that the segmented region be of homogeneous intensity. For example, we may wish to sacrifice the requirement that the region is of homogeneous intensity if we would like the boundary to be very smooth, in which case some pixels will be included in the segmentation region which can be of very varied intensity. We have no such ideas of convexity and the existence of a solution, or freedom to trade-off certain

properties, with the "black-box" algorithms which can give vastly different segmentation results for only a slight change in the initial conditions. This motivates our pursual of the variational approach.

The minimisation of an energy functional results in a partial differential equation (PDE) which we must solve. The first decision we must make is how we should solve this PDE. It is continuous but typically has no analytical solution. Therefore, we solve a discretised formulation of this PDE and have a wide variety of algorithms we can use to solve the resulting non-linear equations. In this thesis, we focus on using the multigrid algorithm which has optimal complexity, i.e. the computational complexity increases linearly with the number of variables we must solve for. In this thesis we find that for highly non-linear PDEs, which typically result from minimising image segmentation models, the multigrid method performs poorly and does not converge. We perform a detailed analysis and find that the multigrid smoothing schemes, which are supposed to effectively reduce the algebraic errors in the solution, perform poorly for pixels near edges in images. Thereforre, we present two new smoothing schemes which are designed to effectively damp the algebraic errors at edge pixels specifically and perform well on all other pixels. This gives rise to a convergent multigrid method.

In Chapter 2, supplemented by Appendix A, we give all the background information required for Chapters 3, 4 and 5 which constitute the contribution of this thesis. We will now summarise the contents of each chapter.

Chapter 2. This chapter gives an extensive and thorough background to all the required theory and results for the remaining chapters. This chapter contains established results and is structured in the following way:

- In §2.1 we give an introduction to inverse problems, their relation to imaging problems and regularisation techniques which make ill-posed problems into well-posed problems.
- In §2.2 we give an introduction to the notion of global and selective image segmentation in a variational framework. We also review global and selective image segmentation models that are found in the literature.
- In §2.3 we give an introduction to the theory behind multigrid iterative solvers – both linear and non-linear.

The contribution of this thesis is all contained in Chapters 3, 4 and 5. These have all been published in peer-reviewed journals. We will now detail the content of each chapter and the corresponding paper references; we will also detail the contribution that the author of this thesis made to each of the papers.

Chapter 3. This chapter is largely taken from [137], an accepted and published paper. In this chapter, we develop a new convex selective segmentation model with a modified geodesic distance penalty term. We now regard the distance between pixels, not in the Euclidean sense, but dependent on the intensities of the pixels on the path joining the pixels. For example, two pixels may be far from one another in a Euclidean sense, but if there is a path between them along which all pixels have a similar intensity, then the geodesic distance between these pixels is small. We can obtain accurate segmentation results which other state-of-the-art models struggle to achieve, such as the segmentation of objects in medical scans with blurred boundaries and in the presence of noise. Many results are presented in this chapter. This work was conducted under the supervision of my two supervisors: Prof. Ke Chen and Prof. Klaus Irion. The key ideas were mine, the experiments were all performed by me and the manuscript was written by myself with supervisor review.

Chapter 4. This chapter is based on [140], an accepted and published paper. In this chapter, we develop a new convex selective image segmentation model with a new intensity fitting term. This allows for the segmentation of images with very similar average foreground and background intensities. For existing models, the results are poor or the models are not robust to input parameters. Our model gives excellent results on a wide variety of images and is very robust to the main parameters compared with competitor models. This work was conducted under the supervision of Dr. Jack Spencer. The key ideas were mine and the experiments were performed by myself. I also wrote the vast majority of the manuscript, Dr. Spencer performed a supervisory role giving guidance on the best way to present the manuscript. He also provided the images in Figure 4.21 and the benchmark data in Figures 4.20 and 4.22.

Chapter 5. This chapter is based on [138], an accepted and published paper. We focus on developing a convergent non-linear multigrid algorithm by developing a non-standard novel smoother. It is found that standard smoothers, such as Jacobi and Gauss-Seidel, are ineffective at pixels near edges in images. Therefore, we perform a detailed analy-

sis to find the reason behind this using Local Fourier Analysis. We then develop two new smoothers which are effective at damping the errors at these edge pixels and perform well at all other pixels. This work gives rise to a convergent non-linear multigrid algorithm and we demonstrate its applicability to two selective segmentation models, namely the Rada-Chen model [135] and the Spencer-Chen model [153]. We find that further work will be necessary to improve the stability of the smoothers to permit us to apply multigrid to the convex relaxed models we have developed and detailed in Chapters 3 and 4 and hence they are not considered here. Discussion of the detailed reasons behind this instability is given in the chapter. This work was conducted under the supervision of my two supervisors: Prof. Ke Chen and Prof. Klaus Irion. The key solutions to problems presented are mine and I performed the experiments. I also wrote the manuscript with supervisor review.

Chapter 2

Preliminaries

2.1. Inverse Problems

An inverse problem is a problem of using observed measurements to infer the input data or parameter values of a physical model. In contrast to a direct modelling problem, where we try to find exact or approximate results which describe various phenomena using some input data, the inverse problem uses those results as its starting point and tries to determine the unknown input data and initial parameters.

2.1.1. Hadamard

Definition 2.1.1.1 (Well-posed Problem)**.** According to Hadamard [85], a problem is well-posed (or correctly-set) if:

1. the solution exists,
2. the solution is unique,
3. the solution depends continuously on the data and parameters.

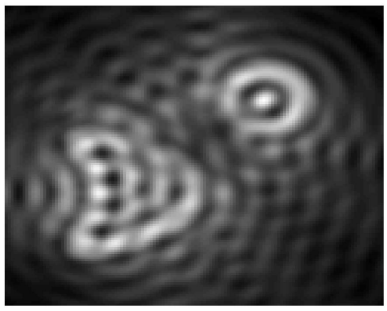
Condition (2) means that the solution is "unique within a certain class of functions". For example, a problem which has two solutions, one of which is in the vector space X and another in the vector space Y can still be well-posed in the space X or Y. Condition (3)

means that a "small" perturbation in initial data or in the parameter values results in a "small" change in the solution (in some appropriate norm). If the problem is well-posed, in a space, then it is potentially possible to compute a solution to this problem using a stable algorithm.

Definition 2.1.1.2 (Ill-posed Problem.)**.** Problems that are not well-posed in the sense of Hadamard are termed ill-posed.

Often, the ill-posedness of certain practical problems is due to their lack of precise mathematical formulation. Ill-posed problems are usually understood to be those problems where a small change in the initial data or parameters leads to an arbitrarily large change in the output data. Inverse problems are typically ill-posed, with the stability condition of well-posedness most often violated. All of the key applications of image processing present us with inverse problems, such as segmentation, registration, deblurring, denoising, dehazing, enhancement, restoration, optical flow, video tracking, etc. In these cases, we are provided with an image or a set of images, and attempt to retrieve certain properties or features from the image.

Example 2.1.1.3. Suppose we have an object or objects in a domain. We wish to determine the shape of those objects by measuring the scattering effect the objects have on different frequency waves directed at the objects at various angles. In Figure 2.1, we give an example of a typical scattering image. The ill-posed inverse problem is to determine how many objects we have in the image and the shape of these objects. It is ill-posed as we have multiple possible solutions due to the boundaries of the objects being unclear.



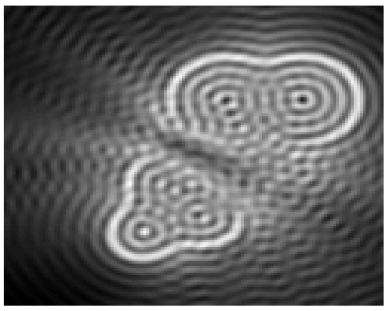
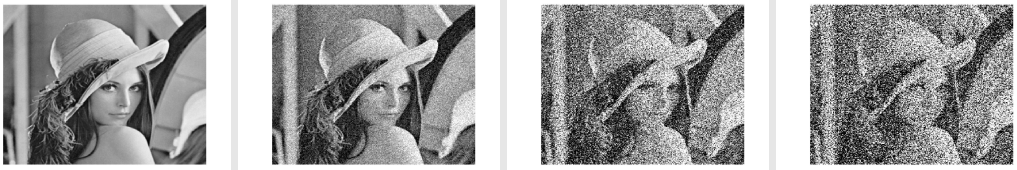


Figure 2.1: Examples of scattering results, provided by Yukun Guo (Harbin, China).

2.1.2. Ill-Posed Imaging Problems

Within image processing, there are many ill-posed inverse problems such as denoising, deblurring and segmentation. We give examples of these below. In this thesis, our focus will be on the problem of image segmentation. We will discuss this more formally in §2.2.

Example 2.1.2.1 (Denoising)**.** Suppose that we are provided with an image corrupted by noise. The aim of *denoising* is to remove the noise and recover the original *clean* image. In (b)–(d), we have images corrupted with different noise levels; the denoised image would more closely resemble the image in (a).



(a) Noise Free Image (b) 1% Noise (c) 10% Noise (d) 20% Noise

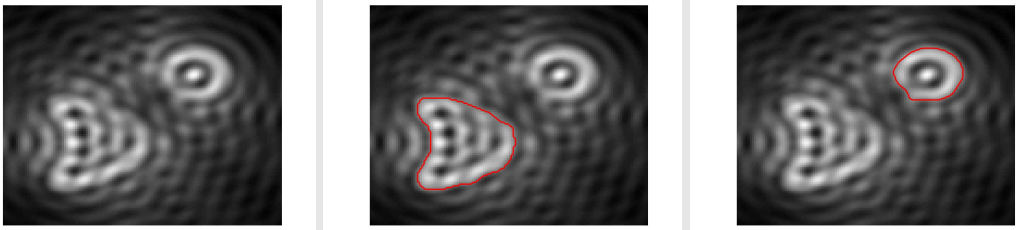
Figure 2.2: The classic Lena test image. In (a) we show the clean image and in (b)–(d) the image corrupted with various levels of Gaussian noise (by convention denoted by σ).

Example 2.1.2.2 (Deblurring)**.** Suppose that we are provided with an image corrupted by blur. The aim of *deblurring* is to remove the blur and recover the original *clean* image. In (b)–(d), we have images corrupted with different levels of Gaussian blur; the deblurred image would more closely resemble the image in (a).

(a) Blur Free Image (b) $\sigma = 1$ (c) $\sigma = 2$ (d) $\sigma = 5$

Figure 2.3: The classic Cameraman test image. In (a), we show the clean image and in (b)–(d), the image corrupted with various levels of Gaussian blur.

Example 2.1.2.3 (Segmentation)**.** Suppose that we are provided with an image and we would like to segment out the objects in the foreground. This is the task of finding the boundary of the object or objects. For example, in the images below we would like to segment the two objects in image (a). In (b) and (c) we give two segmentation results for objects in the image.



(a) Original Image (b) Segmentation 1 (c) Segmentation 2

Figure 2.4: In (a), we have the original image, and in (b) and (c) the segmentation results for each object in the image. The red line represents the boundary of the object.

2.1.3. Regularisation

Regularisation of an inverse problem is the technique of making a potentially ill-posed problem well-posed. The problems we consider in this thesis are ill-posed minimisation problems for which Tikhonov et al. [157] introduced a popular regularisation technique. Their idea is to include a constraint term in the ill-posed minimisation problem which encourages the solution to be a member of a specified set $\mathcal{S}$. Examples for the set $\mathcal{S}$ are: continuous functions, smooth functions and functions taking values in $[a, b]$ for $a, b \in \mathbb{R}$.

Example 2.1.3.1. Suppose we are looking for the solution $\boldsymbol{x}$ of

$$A\boldsymbol{x} = \boldsymbol{b}$$

with

$$A = \begin{bmatrix} 0.16 & 0.10 \\ 0.17 & 0.11 \\ 2.02 & 1.29 \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} 0.27 \\ 0.25 \\ 3.33 \end{bmatrix}.$$

There is no exact solution $\boldsymbol{x}$ for this problem, however, we can compute a good approximation $\boldsymbol{x}^*$ by solving

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \left\{ ||A\boldsymbol{x} - \boldsymbol{b}||_2^2 \right\},$$

this is called the Least Squares solution. We obtain the solution

$$\boldsymbol{x}^* = \begin{bmatrix} 7.01 \\ -8.40 \end{bmatrix}$$

We note however that

$$A\boldsymbol{x} = \boldsymbol{b} = A \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0.01 \\ -0.03 \\ 0.02 \end{bmatrix}$$

and we desire the solution near $[1, 1]^T$. If we use Tikhonov regularisation to penalise the norm of $\boldsymbol{x}^*$ we obtain the following minimisation problem

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \left\{ ||A\boldsymbol{x} - \boldsymbol{b}||_2^2 + ||\boldsymbol{x}||_2^2 \right\},$$

which has solution

$$\boldsymbol{x}^* = \begin{bmatrix} 0.99 \\ 0.64 \end{bmatrix}.$$

This is far better than the least squares solution, however is still far from the desired solution. We can impose the known solution using the regularisation term in the following minimisation

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \left\{ ||A\boldsymbol{x} - \boldsymbol{b}||_2^2 + \left|\left| \boldsymbol{x} - [1, 1]^T \right|\right|_2^2 \right\},$$

which penalises the solution $\boldsymbol{x}^*$ being too far from $[1, 1]^T$. Performing this minimi-

sation, we obtain the solution

$$\boldsymbol{x}^* = \begin{bmatrix} 1.01 \\ 1.00 \end{bmatrix}.$$

In the context of imaging, we frequently use regularisation in our models to ensure the solutions have desired properties. One crucial regulariser used extensively throughout imaging is the Total Variation (TV) regulariser. This was introduced in the seminal paper of Rudin, Osher and Fatemi (ROF) in [144] in the context of denoising images. In image processing, we often constrain our solutions to the set of functions with bounded variation, i.e. with finite total variation.

Definition 2.1.3.2 (Total Variation)**.** Let Ω be a bounded open set in $\mathbb{R}^d$ with C^1 boundary $\partial\Omega$. For L^1 function $u : \Omega \to \mathbb{R}$ with $\boldsymbol{x} \in \Omega$, the total variation is given by

$$TV(u;\Omega) = \int_\Omega |\nabla u(\boldsymbol{x})| \, \mathrm{d}\Omega, \tag{2.1}$$

which we may also denote $TV(u)$ when the domain is clear.

Definition 2.1.3.3 (Bounded Variation)**.** The set of functions with bounded variation on domain Ω is defined

$$BV(\Omega) = \{u \in L^1(\Omega) \mid TV(u;\Omega) < \infty\}$$

There are many potential regularisers also used in image processing, such as Total Generalised Variation [30] and Euler Elastica [122, 152, 156], however, in this thesis, we will not venture much beyond considering the TV regulariser as it performs well in our models.

2.2. Variational Image Segmentation Models

In this section, we will discuss the notion of image segmentation and its applications. We will focus on the variational approach to image segmentation. This is a highly mathematical viewpoint from which to address the problem (in contrast to the simple algorithmic approaches that can be taken, e.g. image thresholding). This variational formulation

allows us to utilise robust and rigorous mathematical techniques to prove, for example, that some models have unique segmentation results and that algorithms will give the same results regardless of the initial conditions. These cannot be necessarily be guaranteed by more algorithmic "black box" approaches.

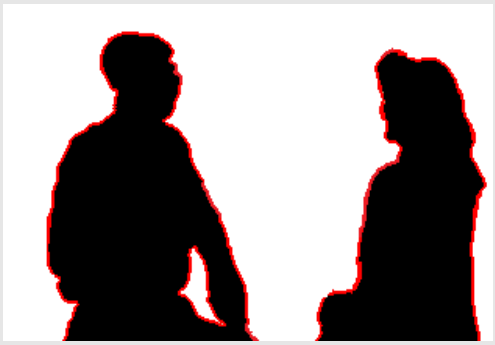
2.2.1. Introduction to Image Segmentation

Image segmentation is the partitioning of an image into different regions. This can simply be the identification of foreground and background in an image (two-phase segmentation) or the separation of an image into multiple regions (multi-phase segmentation).

Example 2.2.1.1 (Two-phase and multi-phase segmentation)**.**

Two-phase segmentation.

(a) Original Image.



(b) Two-phase segmentation result.

Multi-phase segmentation

(c) Original Image.

(d) Multi-phase segmentation result.

Figure 2.5: Examples of two-phase and multi-phase segmentation results.

2.2.2. Representing the Segmentation Boundary

Typically, in image segmentation, the final image segmentation is achieved after an iterative process; we start with some general initialisation and run the algorithm until the segmentation boundary has stopped moving. For the two-dimensional examples of segmentation results in Figures 2.5, 2.6, 2.7, 2.10 and 2.11, the segmentation boundary (shown in red) is a one-dimensional line generally denoted Γ in the literature.

This Γ can be parametrised for the initialisation and subsequently updated with each iteration. However, we find that topology changes (splitting or merging of the boundary) are extremely complex to encode by parametrisation. One simple and novel technique first introduced by Dervieux and Thomasset [59, 60] and popularised by Osher and Sethian [125] is to embed Γ in higher dimensional space. This is popularly known as the level set method and now, rather than track a parametrised Γ, we instead track the zero level set of a function ϕ.

Example 2.2.2.1 (Representing the segmentation boundary as a level set)**.**

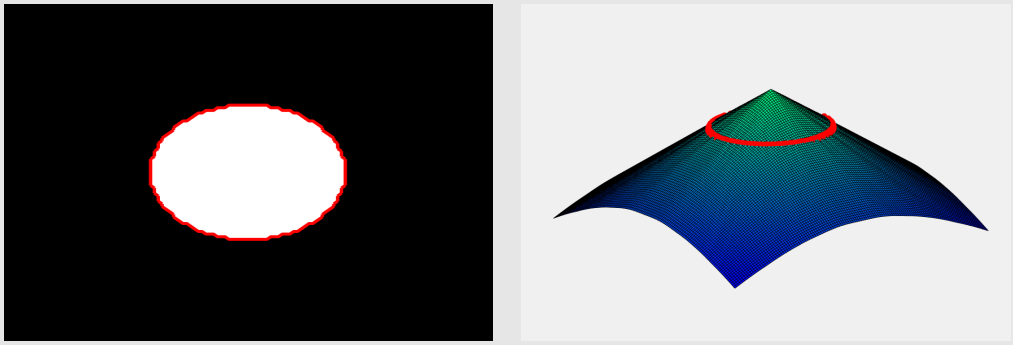


Figure 2.6: One object and the embedding of its boundary as a level set.

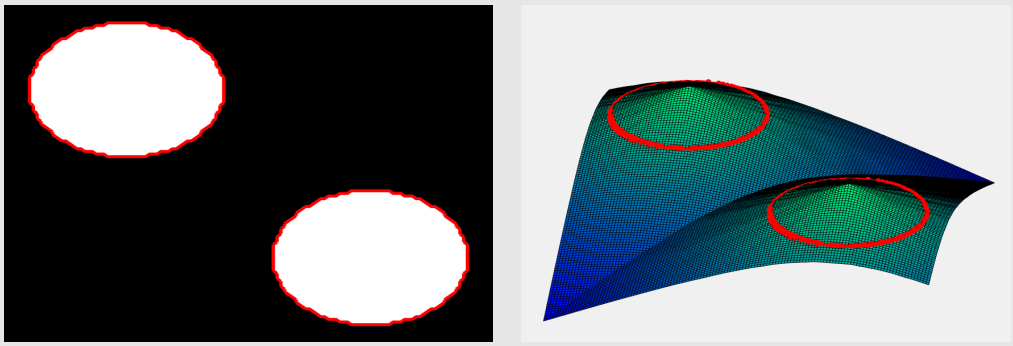


Figure 2.7: Two objects and the embedding of the boundary as a level set.

The level set method is very successful as the tracking of topology changes is easy. We are now focused on finding the function ϕ and the object boundary Γ is indirectly obtained as

$$\Gamma = \{\boldsymbol{x} \in \Omega \mid \phi(\boldsymbol{x}) = 0\}.$$

We can take the idea further and determine which pixels are foreground and which are background by

$$\begin{cases} \text{Foreground} = \{\boldsymbol{x} \mid \phi(\boldsymbol{x}) > 0\} \\ \text{Background} = \{\boldsymbol{x} \mid \phi(\boldsymbol{x}) < 0\} \end{cases}$$

and using the Heaviside function

$$H(\phi) = \begin{cases} 1, & \phi \geq 0, \\ 0, & \phi < 0, \end{cases} \tag{2.2}$$

we can now characterise the foreground and background by

$$\begin{cases} \text{Foreground} = \{\boldsymbol{x} \mid H(\phi(\boldsymbol{x})) = 1\} \\ \text{Background} = \{\boldsymbol{x} \mid H(\phi(\boldsymbol{x})) = 0\} \end{cases}$$

The application of level set methods to a wide variety of problems including image processing, fluid dynamics flows, visualisation, computer vision, control, visibility, segmentation, restoration and many others can be found in [115, 124, 149, 150].

2.2.3. Variational Approach

In this thesis, we study variational image segmentation. The variational approach is the study of minimising energy functionals, i.e. problems of the form

$$\Phi = \arg\min_{\phi \in \mathcal{S}} \mathcal{F}(\phi)$$

where Φ is the optimiser of $\mathcal{F}$ and Φ lies in the space $\mathcal{S}$. We have the freedom to design $\mathcal{F}$, and choose the space $\mathcal{S}$, such that the optimiser has the desired properties for our problem. For example, in image segmentation, we can design a functional $\mathcal{F}$ such that the optimiser (the segmentation result) has desirable properties. Examples include: a smooth boundary between regions, and a short boundary between regions, the segmented regions have homogeneous intensity, etc. A typical image segmentation functional is of the form

$$\mathcal{F}(\phi) = \mu \underbrace{\int_\Omega \mathcal{R}(\phi)\, \mathrm{d}\Omega}_{\text{Regulariser}} + \lambda \underbrace{\int_\Omega \mathcal{H}(\phi)\, \mathrm{d}\Omega}_{\text{Data Fitting Term}}$$

where μ and λ are real non-negative parameters used to tune the final solution by varying the weighting of the contribution from the regulariser term and the data term. A

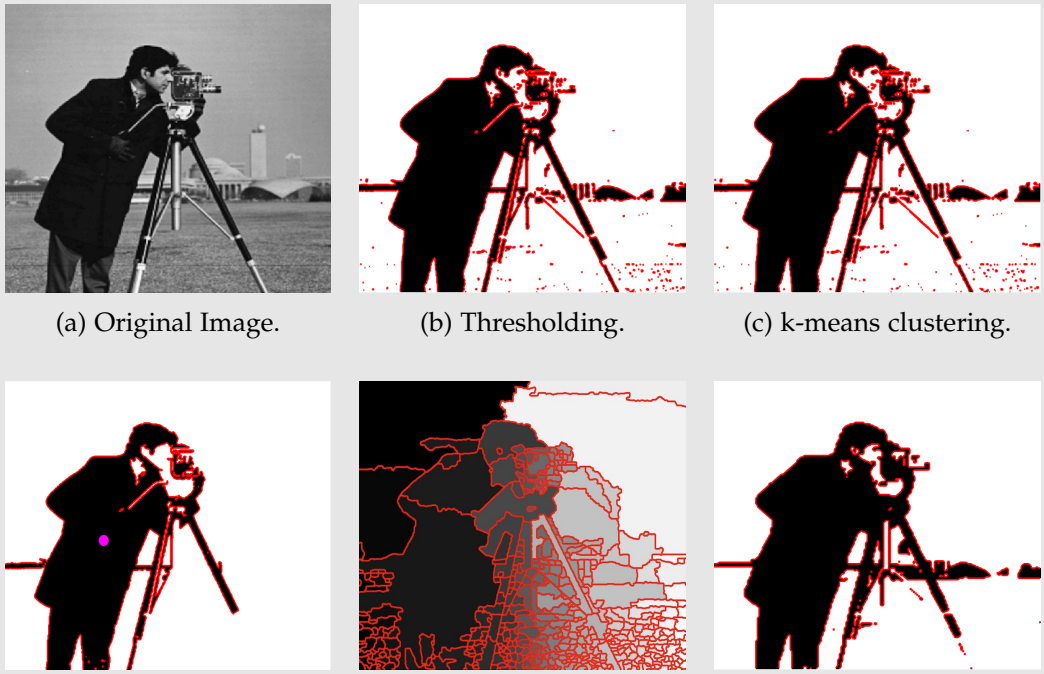
large μ value gives a more regular solution and a large λ gives a solution which fits the data more closely. We aim to find the minimiser of $\mathcal{F}(\phi)$ by finding its turning point. We do this by finding the value of $\phi \in \mathcal{S}$ where the first variation (§A.2.1) is zero, i.e.

$$\delta\mathcal{F}(\phi;\psi) = 0$$

for arbitrary $\psi \in \mathcal{S}$. The design and analysis of these functionals and finding the optimiser in an image segmentation context is what we call "variational image segmentation". Approaches to image segmentation problems (both global and selective) broadly fall into two classes; region-based and edge-based.

Region-Based Segmentation. This type of image segmentation relies on the assumption that neighbouring pixels in an image have the same or similar intensity. There are many algorithms which can be used which tend to fall into two categories. The first category is simple algorithms requiring black box operations to obtain the result. These include such algorithms as thresholding, k-means clustering [114], seeded region growing [1] and watershed [163] which typically require minimal user input and are generally automatic. The second category of region-based segmentation algorithms require more complex mathematics to solve. These can include a vast array of numerical methods and require much more user input. Examples include the variational models of Mumford-Shah [119] and Chan-Vese [46] discussed later in § and § respectively. As we will see later in Chapters 3 and 4, the segmentation result is only attained after solving a nonlinear PDE, which introduces numerical problems itself, and the user must choose some appropriate parameters in the model itself, otherwise, the results can be undesirable.

Example 2.2.3.1 (Region-based segmentation).



(a) Original Image. (b) Thresholding. (c) k-means clustering.

(d) Region growing (seed indicated in pink). (e) Watershed. (f) Chan-Vese.

Figure 2.8: Results for various region-based image segmentation algorithms.

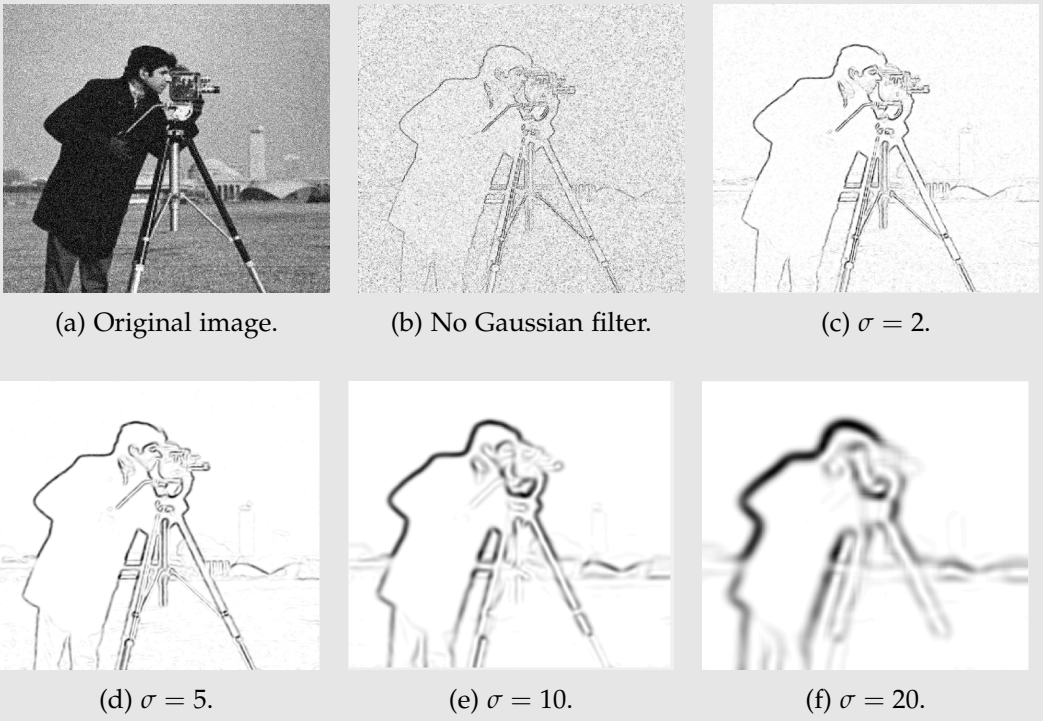
Edge-Based Segmentation. This type of image segmentation assumes that there is an edge between different regions in an image. The edge-based algorithms will then find the segmentation for which the boundaries between regions lie on the edges in the image. Typically, we first must find the edges in the image. Let us first suppose the image is denoted by z and is continuous. Throughout the imaging literature [35, 36, 42, 78] the most commonly used edge detector function is

$$g\left(|\nabla z|\right) = \frac{1}{1+\beta\left|\nabla\left(G_\sigma * z\right)\right|^2} \tag{2.3}$$

which is close to zero at edges and $g \approx 1$ away from them. G_σ is a Gaussian filter with parameter σ; this is convolved with the image z and blurs it. This convolution blurs the noisy pixels in the image and ensures that noise doesn't distract from the edges in

the image. In Figure 2.9, we give the edge detector for varying σ values. If we use no Gaussian convolution, the noise in the image results in an unclear edge map. However, with even a small filter ($\sigma = 2$) we achieve a good edge map but, as σ grows large, the map becomes inaccurate. Typically, we must vary σ dependent on the level of noise in the image provided.

Example 2.2.3.2 (Edge-detector for varying σ)**.**



(a) Original image. (b) No Gaussian filter. (c) $\sigma = 2$.

(d) $\sigma = 5$. (e) $\sigma = 10$. (f) $\sigma = 20$.

Figure 2.9: Edge detector (2.3) for varying σ. In (b), we show the result for no Gaussian filtering of the image.

One simple approach to edge-based segmentation is just to use the output from the edge detector function as our segmentation result by thresholding it. In Figure 2.9(d), we see that this gives a good outline of the objects in the image and segments the image into many separate domains. More complex mathematical models have been derived based on edge detection in the variational framework; the earliest work being the seminal "snakes" model of Kass et al. [97]. This was further developed by Casselles et al. who introduced another seminal model, the Geodesic Active Contour (GAC) model [36].

The state-of-the-art models within variational image segmentation, both global and selective [57, 98, 121, 137, 153], aim to fuse the benefits of region-based and edge-based segmentation. The models we will study from this point onwards are all variational models.

2.2.4. Convex and Non-Convex Variational Models

Definition 2.2.4.1. In this thesis, we call the minimisation of a given functional a "model" and each model is given a name to indicate who invented or proposed it.

Example 2.2.4.2. The famous Rudin, Osher and Fatemi (ROF) [144] denoising model for image z is given by

$$u^* = \arg\min_{\phi \in BV(\Omega)} \mathcal{F}(u)$$

where

$$\mathcal{F}(u) = \mu \int_\Omega |\nabla u| \, \mathrm{d}\Omega + \lambda \int_\Omega ||u - z||_2^2 \, \mathrm{d}\Omega$$

Definition 2.2.4.3. A model is convex or non-convex if the functional to be minimised is convex or non-convex respectively.

Theorem 2.2.4.4. If a convex model has a minimiser, it must be the global minimiser.

Proof [20] Suppose the model is given as

$$u^* = \arg\min_{u \in \mathcal{S}} \mathcal{F}(u)$$

and $\mathcal{F}$ is given. Suppose that u^* is a local minimum of $\mathcal{F}$ over $\mathcal{S}$, it follows that there exists $r > 0$ such that $\mathcal{F}(u) \geq \mathcal{F}(u^*)$ for any $u \in \mathcal{S}$ satisfying $u \in B(u^*, r)$. Now let $v \in \mathcal{S}$ satisfy $v \neq u^*$. Our objective is to show that $\mathcal{F}(v) \geq \mathcal{F}(u^*)$. Let $\lambda \in (0, 1]$ be such that $u^* + \lambda(v - u^*) \in B(u^*, r)$. An example of such λ is $\lambda = \frac{r}{||v-u^*||}$. Since $u^* + \lambda(v - u^*) \in B(u^*, r) \cap \mathcal{S}$, it follows that $\mathcal{F}(u^*) \leq \mathcal{F}(u^* + \lambda(v - u^*))$, and hence by

Jensen's inequality

$$\mathcal{F}(u^*) \leq \mathcal{F}(u^* + \lambda(v - u^*)) \leq (1-\lambda)\mathcal{F}(u^*) + \lambda\mathcal{F}(v).$$

Thus, $\lambda\mathcal{F}(u^*) \leq \lambda\mathcal{F}(v)$, and hence the desired inequality $\mathcal{F}(u^*) \leq \mathcal{F}(v)$ follows. □

2.2.5. Global and Selective Image Segmentation

Image segmentation can be performed in two distinct ways, each with different applicability. Firstly, global segmentation is the isolation of all objects in an image from the background and secondly, selective segmentation is the isolation of a subset of the objects in an image from the background. Selective segmentation is very useful in, for example, medical imaging for the segmentation of single organs.

Note 2.2.5.1. The segmentation results can be presented in two ways, the first being as in Figure 2.5 where we give a cartoon image and mark the boundary between regions in red. The alternative, preferred in this thesis, will be the presentation in Figures 2.10 and 2.11, where we overlay the region boundary over the original image. The boundary will always be given as a red line unless stated otherwise.

Example 2.2.5.2 (Global and selective segmentation)**.**

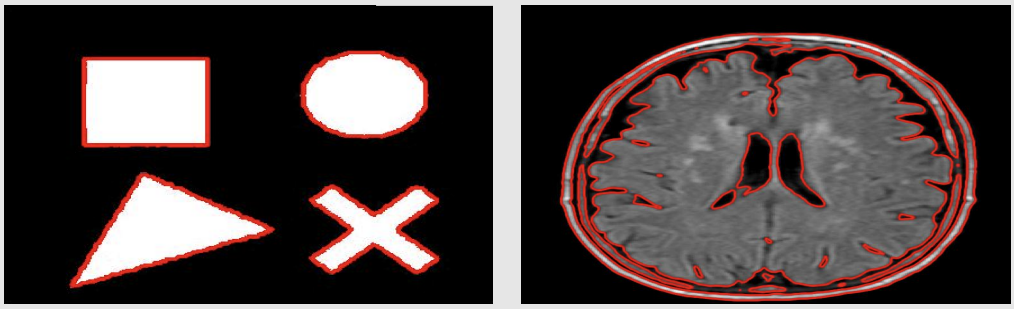


Figure 2.10: Example global segmentation results.

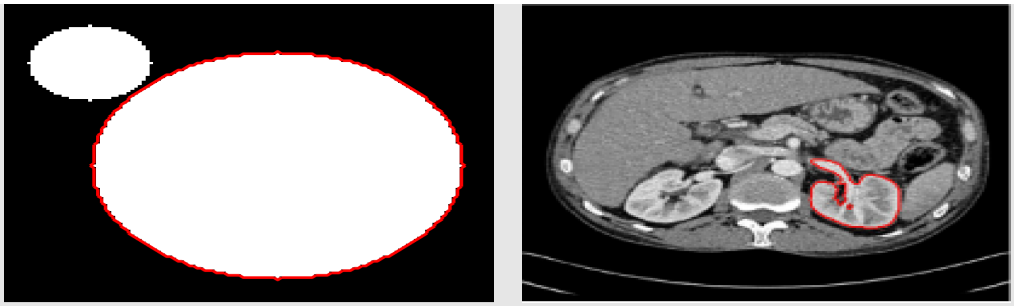


Figure 2.11: Example selective segmentation results.

2.2.6. Global Image Segmentation

Variational image segmentation involves finding the boundary Γ of an object by minimising an energy functional with respect to some variable, generally denoted ϕ for non-convex models and u for convex models.

In the case of non-convex image segmentation models, the functional is designed in such a way that the level set $\Gamma = \{\phi = 0\}$ lies on the boundary of the objects in the image. The inside of the objects is given by $\{\phi > 0\}$ and the outside by $\{\phi < 0\}$. The exact values of ϕ are not of interest, we focus only on the zero level set and the regions that are positive and negative. Alternatively, using the Heaviside function

$$H(\phi) = \begin{cases} 1, & \phi \geq 0, \\ 0, & \phi < 0, \end{cases}$$

the inside of the objects is characterised by the Heaviside taking value 1.

For the convex (relaxed) models we see later in this chapter and in Chapters 3 and 4, the minimisation with respect to u leads to an almost binary function u which takes value 1 inside the objects and 0 elsewhere. Using a result in [44], the global minimiser of the energy functional is found for almost every $\gamma \in (0, 1)$ by setting $u^* = \chi_\Sigma$, where χ is the characteristic function and $\Sigma = \{\boldsymbol{x} \mid u(\boldsymbol{x}) > \gamma\}$. In all tests in this thesis, we set $\gamma = 0.5$. The boundary of the object or objects is then given as the level set of u^* at any value in $(0, 1)$.

We will initially discuss the non-convex Mumford-Shah [119] and Chan-Vese [46] global

segmentation models. We will conclude the discussion of global segmentation models by reviewing the convex relaxation technique of Chan et al. [44] which allows reformulation of the non-convex Chan-Vese model to a convex model with a unique minimiser.

2.2.6.1 Mumford-Shah Model (1989)

The model of Mumford and Shah [119] is one of the most famous and important variational models in image segmentation. This aims to find a piecewise smooth approximation to an image z. The model is given by

$$(\phi^*, \Gamma^*) = \arg\min_{\phi,\Gamma} \mathcal{F}_{MS}(\phi, \Gamma)$$

where

$$\mathcal{F}_{MS}(\phi, \Gamma) = \mu \int_\Gamma \mathrm{d}s + \lambda \int_\Omega (z-\phi)^2 \,\mathrm{d}\Omega + \int_{\Omega\setminus\Gamma} |\nabla\phi|^2 \,\mathrm{d}\Omega$$

with Γ being the set of discontinuities in ϕ. Each term in this functional aims to enforce specific properties in the solution ϕ. Namely, the first term ensures that the overall length of the boundary between the piecewise smooth regions is short. The second term encourages the solution to be close to the original image z and the final term encourages ϕ to be smooth away from the discontinuity set. In Figure 2.12, we see the Mumford-Shah result for an image in [132]. The result is a piecewise smooth image and the set Γ segments the domain into different regions.

Example 2.2.6.1 (Mumford-Shah Model)**.**

(a) Original Image.

(b) ϕ^* obtained by the Mumford-Shah model.

Figure 2.12: Example of Mumford-Shah result from [132]

Computing the minimiser of $\mathcal{F}_{MS}$ is very challenging due to the non-regularity of Γ. Ambrosio and Tortorelli [6] design an approximation to $\mathcal{F}_{MS}$ by using a sequence of simpler elliptic variational problems. Pock et al. [132] use convex relaxation techniques to minimise $\mathcal{F}_{MS}$ and a level set approach has also been used [47, 159].

A simpler variant of $\mathcal{F}_{MS}$ would be to assume only that the regions of the solution are piecewise constant, i.e. we assume that $\phi = c_k$ on Ω_k, where $\Omega = \bigcup_k \Omega_k$. This piecewise constant model is given by

$$(\phi^*, \Gamma^*) = \arg\min_{\phi,\Gamma} \mathcal{F}_{PCMS}(\phi, \Gamma)$$

where

$$\mathcal{F}_{PCMS}(\phi, \Gamma) = \mu \int_\Gamma \, \mathrm{d}s + \sum_k \lambda_k \int_{\Omega_k} (z - c_k)^2 \, \mathrm{d}\Omega_k$$

where z is the given image, μ, λ_k are fixed non-negative real parameters. We have dropped the smoothness term as each region of $\Omega \backslash \Gamma$ is assumed constant. This is a useful model to partition the image into k piecewise constant regions. In this thesis however, we are concerned only with two-phase image segmentation, i.e. the partitioning of an image into foreground and background and $k = 2$.

2.2.6.2 Chan-Vese Model (2001)

In their seminal paper *Active Contours Without Edges* [46], Chan and Vese give a technique for solving the piecewise constant Mumford-Shah model $\mathcal{F}_{PCMS}$ when $k = 2$. Explicitly, the Chan-Vese model is given by

$$(\Gamma^*, c_1^*, c_2^*) = \arg\min_{\Gamma, c_1, c_2} \mathcal{F}_{CV}(\Gamma, c_1, c_2)$$

where

$$\mathcal{F}_{CV}(\Gamma, c_1, c_2) = \mu \int_\Gamma \, \mathrm{d}s + \lambda_1 \int_{\Omega_1} (z - c_1)^2 \, \mathrm{d}\Omega_1 + \lambda_2 \int_{\Omega_2} (z - c_2)^2 \, \mathrm{d}\Omega_2 \tag{2.4}$$

where z is the given image and $\Omega = \Omega_1 \cup \Omega_2$ with μ, λ_1 and λ_2 fixed non-negative real parameters. The values c_1 and c_2 are the average intensities of z inside Ω_1 and Ω_2 respectively. In §2.2.2, we discussed that the tracking of Γ is extremely difficult and that one solution is to embed it in a higher dimension function ϕ which has the property that $\Gamma = \{\boldsymbol{x} \mid \phi(\boldsymbol{x}) = 0\}$.

Let us define ϕ such that

$$\phi(\boldsymbol{x}) \begin{cases} > 0, & \boldsymbol{x} \in \Omega_1, \\ < 0, & \boldsymbol{x} \in \Omega_2, \\ = 0, & \boldsymbol{x} \in \Gamma. \end{cases}$$

We now aim to express the first term of $\mathcal{F}_{CV}$, the length of Γ in terms of ϕ. This can be accomplished by the relation

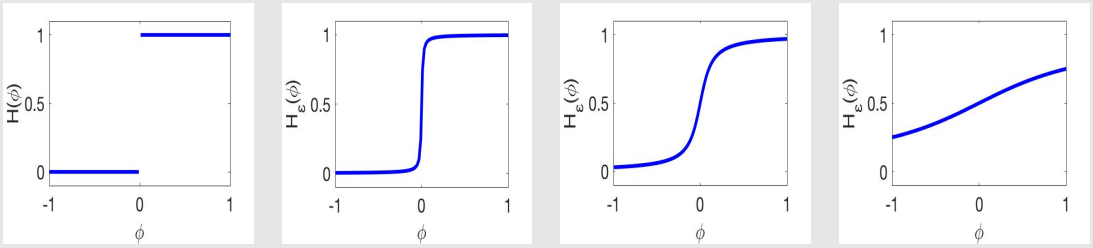
$$\int_\Gamma \mathrm{d}s = \int_\Omega |\nabla H(\phi)| \, \mathrm{d}\Omega$$

where $H(\phi)$ is the Heaviside function defined in (2.2). However, as we must find $\nabla H(\phi)$ but $H(\phi)$ is discontinuous at zero, we must regularise $H(\phi)$ at the discontinuity. In this thesis we use

$$H_\varepsilon(\phi) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{\phi}{\varepsilon}\right)\right) \tag{2.5}$$

with small ε. In Figure 2.13, we show how this function approximates $H(\phi)$ more accurately for small ε. Alternative regularisations are possible for $H_\varepsilon(\phi)$, see [45] for several options.

Example 2.2.6.2 (Regularised Heaviside function)**.**



(a) $H(\phi)$ (b) $H_\varepsilon(\phi)$, $\varepsilon = 0.01$. (c) $H_\varepsilon(\phi)$, $\varepsilon = 0.1$. (d) $H_\varepsilon(\phi)$, $\varepsilon = 1$.

Figure 2.13: Heaviside function and $H_\varepsilon(\phi)$ for varying ε.

We also note that the second and third terms of the Chan-Vese functional $\mathcal{F}_{CV}$ can be rewritten using Heaviside functions

$$\lambda_1 \int_\Omega (z-c_1)^2 H(\phi)\ \mathrm{d}\Omega + \lambda_2 \int_\Omega (z-c_2)^2\,(1-H(\phi))\ \mathrm{d}\Omega$$

which removes consideration of Ω_1 and Ω_2 completely. Overall, the Chan-Vese model can be written in the level set framework as

$$(\phi^*, c_1^*, c_2^*) = \arg\min_{\phi, c_1, c_2} \mathcal{F}_{CV}(\phi, c_1, c_2)$$

where

$$\begin{aligned}\mathcal{F}_{CV}(\phi, c_1, c_2) = \mu \int_\Omega |\nabla H_\varepsilon(\phi)| d\Omega + \lambda_1 \int_\Omega (z-c_1)^2 H_\varepsilon(\phi)\ \mathrm{d}\Omega \\ + \lambda_2 \int_\Omega (z-c_2)^2 (1-H_\varepsilon(\phi))\ \mathrm{d}\Omega\end{aligned} \tag{2.6}$$

for a given image z.

As the minimisation is performed over three variables (ϕ, c_1, c_2) we perform the minimisations in sequence. Below, we will initially determine the minimising value of $\mathcal{F}_{CV}$ for c_1 and c_2, as these are trivial, and then give the equation for ϕ which minimises the functional.

Minimisation with c_1. Fixing ϕ and c_2, the Gâteaux derivative of $\mathcal{F}_{CV}(\phi, c_1, c_2)$ with respect to c_1 is

$$\begin{aligned}&\lim_{\varepsilon\to 0} \frac{\mathcal{F}_{CV}(\phi, c_1+\varepsilon\psi, c_2) - \mathcal{F}_{CV}(\phi, c_1, c_2)}{\varepsilon} = \\ &\lim_{\varepsilon\to 0} \frac{1}{\varepsilon}\left\{ \lambda_1 \int_\Omega (z-(c_1+\varepsilon\psi))^2 H_\varepsilon(\phi)\ \mathrm{d}\Omega - \lambda_1 \int_\Omega (z-c_1)^2 H_\varepsilon(\phi)\ \mathrm{d}\Omega \right\},\end{aligned} \tag{2.7}$$

we simplify the right-hand side and set to zero

$$\lim_{\varepsilon\to 0} \frac{1}{\varepsilon}\left\{ \lambda_1 \int_\Omega \left[(z-(c_1+\varepsilon\psi))^2 - (z-c_1)^2\right] H_\varepsilon(\phi)\ \mathrm{d}\Omega \right\} = 0, \tag{2.8}$$

$$\lim_{\varepsilon\to 0}\frac{1}{\varepsilon}\left\{\lambda_1\int_\Omega\left[(z-c_1)^2-2(z-c_1)\varepsilon\psi+\varepsilon^2\psi^2-(z-c_1)^2\right]H_\varepsilon(\phi)\,\mathrm{d}\Omega\right\}=0, \tag{2.9}$$

$$\left\{\lambda_1\int_\Omega\left[-2(z-c_1)\psi\right]H_\varepsilon(\phi)\,\mathrm{d}\Omega\right\}=0, \tag{2.10}$$

and as ψ is arbitrary, we must have

$$\int_\Omega zH_\varepsilon(\phi)\,\mathrm{d}\Omega-c_1\int_\Omega H_\varepsilon(\phi)\,\mathrm{d}\Omega=0, \tag{2.11}$$

and finally

$$c_1=\frac{\int_\Omega zH_\varepsilon(\phi)\,\mathrm{d}\Omega}{\int_\Omega H_\varepsilon(\phi)\,\mathrm{d}\Omega}$$

Minimisation with c_2. Fixing ϕ and c_1 we minimise with respect to c_2 by setting

$$c_2=\frac{\int_\Omega z\cdot(1-H_\varepsilon(\phi))\,d\Omega}{\int_\Omega(1-H_\varepsilon(\phi))\,d\Omega}.$$

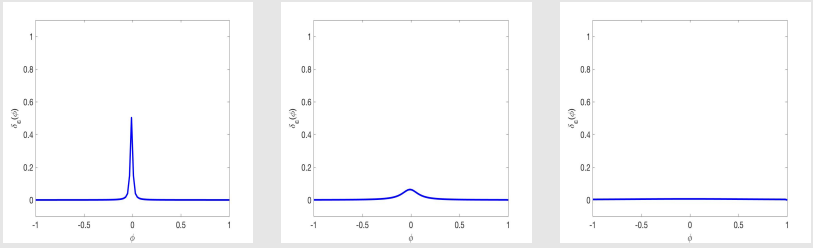
This is obtained in the same manner as the equation for c_1 previously.

Minimisation with ϕ. Fixing c_1 and c_2 we now minimise with respect to the level set function ϕ. Using the Gâteaux derivative and setting to zero, we obtain the following PDE

$$\delta_\varepsilon(\phi)\left\{\mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|_{\varepsilon_1}}\right)-\left[\lambda_1\left(z-c_1\right)^2-\lambda_2\left(z-c_2\right)^2\right]\right\}=0, \tag{2.12}$$

for a given image z with c_1 and c_2 the average intensities for the regions with positive and negative ϕ values respectively. We have $\delta_\varepsilon(\phi)=\frac{\mathrm{d}}{\mathrm{d}\phi}H_\varepsilon(\phi)$ (see Figure 2.14) and we have Neumann boundary condition $\frac{\partial u}{\partial \boldsymbol{n}}=0$ for $\boldsymbol{n}$ the unit outward normal. The denominator of the first term is modified from $|\nabla\phi|$ to $|\nabla\phi|_{\varepsilon_1}=\sqrt{(\phi_x)^2+(\phi_y)^2+\varepsilon_1}$ for small $\varepsilon_1>0$ to avoid numerical problems when $|\nabla\phi|=0$. To find the ϕ which minimises $\mathcal{F}_{CV}(\phi,c_1,c_2)$ we must solve this non-linear PDE for ϕ. We will discuss how we solve this equation in §A.3 and §2.3.

Example 2.2.6.3 (Regularised Delta function)**.**



(a) $\delta_\varepsilon(\phi)$, $\varepsilon = 0.01$. (b) $\delta_\varepsilon(\phi)$, $\varepsilon = 0.1$. (c) $\delta_\varepsilon(\phi)$, $\varepsilon = 1$.

Figure 2.14: Regularised Delta function, $\delta_\varepsilon(\phi) = \mathrm{d}H_\varepsilon(\phi)/\mathrm{d}\phi$ for varying ε.

2.2.6.3 Convex Relaxed Chan-Vese Model (2006)

A drawback of the Chan-Vese energy functional (2.6) is that it is non-convex in ϕ. Therefore, when solving the associated Euler-Lagrange equation (2.12), we may obtain a minimiser ϕ which is only a local minimum of the functional, not the global minimum. Equivalently, we can say that the solution ϕ of (2.12) is dependent on the initialisation used for ϕ in the algorithm.

Example 2.2.6.4 (Results for non-convex Chan-Vese model)**.** We show below how for two different initialisations, we achieve two different results when we use the non-convex Chan-Vese model.

(a) Initialisation 1. (b) Segmentation 1. (c) Initialisation 2. (d) Segmentation 2.

Figure 2.15: Non-Convex Chan-Vese segmentation results for two different initialisations.

This is a serious drawback in the applicability of the segmentation model, as it is not

acceptable to a software user that the segmentation result can be different depending on the initial conditions used in the numerical scheme. Therefore, we prefer to study convex models, where a minimiser is necessarily a global minimiser (Theorem 2.2.4.4).

In Chan et al. [44], the authors address this precise problem, reformulating the non-convex Chan-Vese model to an equivalent convex model. Their first observation is that in the original Chan-Vese paper [46] the authors use the regularised Heaviside function (2.5). Therefore,

$$\delta_\varepsilon(\phi) = \frac{\varepsilon}{\pi\left(\phi^2 + \varepsilon^2\right)}$$

and we note that by definition $\varepsilon > 0$, therefore $\delta_\varepsilon(\phi) > 0$. In which case, we can say that the solution ϕ, which solves the Chan-Vese Euler-Lagrange equation (2.12), is the same as that which solves

$$\mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|_{\varepsilon_1}}\right) - \left[\lambda_1\left(z-c_1\right)^2 - \lambda_2\left(z-c_2\right)^2\right] = 0, \tag{2.13}$$

with Neumann boundary conditions $\frac{\partial\phi}{\partial\boldsymbol{n}} = 0$ for $\boldsymbol{n}$ the unit outward normal.

This is the Euler-Lagrange equation for

$$\mu\int_\Omega |\nabla\phi| d\Omega + \int_\Omega \left[\lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2)\right]\phi \ \mathrm{d}\Omega. \tag{2.14}$$

which is homogeneous in ϕ of degree 1 and $\phi \to \infty$ where $\phi > 0$ and similarly $\phi \to -\infty$ where $\phi < 0$. Therefore this has no minimiser, in general, but an easy fix is to restrict the values ϕ takes to the interval $[0,1]$. With a change of notation, we now solve the following constrained convex minimisation problem

$$\min_{0\le u\le 1}\left\{\mu\int_\Omega |\nabla u| d\Omega + \int_\Omega \left[\lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2\right] u \ \mathrm{d}\Omega\right\}. \tag{2.15}$$

Theorem 2.2.6.5 (Theorem 2 [44])**.** For a given $c_1, c_2 \in \mathbb{R}$ a global minimiser for the Chan-Vese model (2.4) can be found by solving the convex minimisation problem (2.15) and setting $\Gamma = \{\boldsymbol{x} | u(\boldsymbol{x}) > \gamma\}$ for almost every $\gamma \in [0,1]$.

Therefore, we now have a minimisation problem with a unique solution (if it exists) irrespective of the initialisation of u. A drawback of this minimisation problem is the

constraint $u \in [0,1]$ and the authors of [44] make the following claim

Claim 2.2.6.6 (Claim 1 [44])**.** The convex constrained minimisation problem (2.15) has the same set of minimisers as the following convex unconstrained minimisation problem

$$\min_u \left\{ \mu \int_\Omega |\nabla u| d\Omega + \int_\Omega \left[\lambda_1 (z-c_1)^2 - \lambda_2 (z-c_2)^2 \right] u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu(u) \mathrm{d}\Omega \right\},$$

where $\nu(\xi) := \max\{0, 2|\xi - \frac{1}{2}| - 1\}$, provided that

$$\alpha > \frac{\lambda}{2} \left|\left| \lambda_1 (z-c_1)^2 - \lambda_2 (z-c_2)^2 \right|\right|_{L^\infty}.$$

The new term $\nu(u)$ is an exact penalty term [93] which penalises values of u outside the range $[0,1]$, a plot is given in Figure 2.16(a). Explicitly, the convex relaxed model of Chan, Esedoglu and Nikolova [44], is given by

$$(u^*, c_1^*, c_2^*) = \arg\min_{u, c_1, c_2} \mathcal{F}_{CV1}(u, c_1, c_2),$$

where

$$\begin{aligned} \mathcal{F}_{CV1}(u, c_1, c_2) = \mu \int_\Omega |\nabla u| \mathrm{d}\Omega \\ + \int_\Omega \left[\lambda_1 (z-c_1)^2 - \lambda_2 (z-c_2)^2 \right] u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu(u) \mathrm{d}\Omega \end{aligned} \tag{2.16}$$

with the given image z, $\alpha > \frac{\lambda}{2} \left|\left| \lambda_1 (z-c_1)^2 - \lambda_2 (z-c_2)^2 \right|\right|_{L^\infty}$ and $\mu, \lambda_1, \lambda_2$ fixed non-negative real parameters.

Note 2.2.6.7. The functional $\mathcal{F}_{CV1}(u, c_1, c_2)$ is convex for u when c_1 and c_2 are fixed.

The minimisation with respect to c_1 and c_2 is accomplished by setting

$$c_1 = \frac{\int_\Omega zu \, \mathrm{d}\Omega}{\int_\Omega u \, \mathrm{d}\Omega} \qquad c_2 = \frac{\int_\Omega z \cdot (1-u) \, d\Omega}{\int_\Omega (1-u) \, d\Omega}.$$

and the minimisation with respect to u gives rise to the following Euler-Lagrange equa-

tion

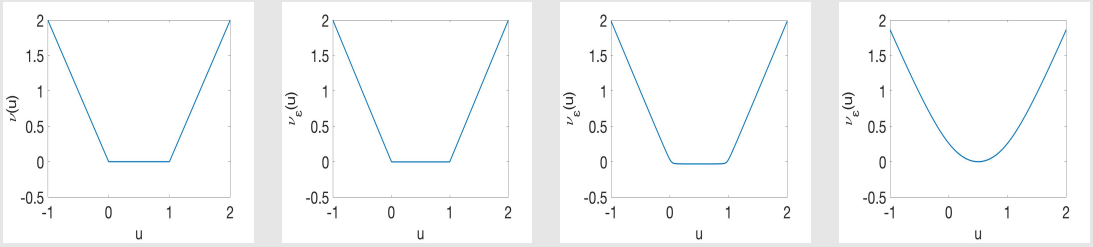
$$\mu\nabla\cdot\left(\frac{\nabla u}{|\nabla u|_{\varepsilon_1}}\right)-\left[\lambda_1(z-c_1)^2-\lambda_2(z-c_2)^2\right]-\alpha\nu'_{\varepsilon_2}(u)=0, \tag{2.17}$$

which we must solve for u with Neumann boundary conditions $\frac{\partial u}{\partial \boldsymbol{n}}=0$ for $\boldsymbol{n}$ the unit outward normal. In this equation, we have introduced two new small regularisation parameters ε_1 and ε_2 to permit the numerical solution of the PDE. The denominator in the first term of (2.17) should be $|\nabla u|$, however, we have numerical problems when $|\nabla u|=0$, therefore we replace it with $|\nabla u|_{\varepsilon_1}=\sqrt{(u_x)^2+(u_y)^2+\varepsilon_1}$ for small $\varepsilon_1>0$. Also, the exact penalty term $\nu(u)$ has kinks at 0 and 1 (see Figure 2.16(a)) but we require the values of it's derivative $\nu'(u)$ and therefore, we use a regularised version from [153]

$$\nu_\varepsilon(u)=H_\varepsilon\left(\sqrt{(2u-1)^2+\varepsilon}-1\right)\left[\sqrt{(2u-1)^2+\varepsilon}-1\right], \tag{2.18}$$

which approximates $\nu(u)$ and $\nu_{\varepsilon_2}(u)\to\nu(u)$ as $\varepsilon_2\to 0$. See the plots in Figure 2.16.

Example 2.2.6.8 (Regularised Penalty function)**.**



(a) $\nu(\phi)$ (b) $\nu_\varepsilon(\phi)$, $\varepsilon=0.01$. (c) $\nu_\varepsilon(\phi)$, $\varepsilon=0.1$. (d) $\nu_\varepsilon(\phi)$, $\varepsilon=1$.

Figure 2.16: Exact penalty function $\nu(u)$ and the regularised version $\nu_\varepsilon(\phi)$, given by (2.18), for varying ε.

The convex relaxed Chan-Vese model permits arbitrary initialisation of u in the PDE (2.17) and we obtain the same result; see Example 2.2.6.9.

Example 2.2.6.9 (Results for convex relaxed Chan-Vese model)**.** We show below how for two different initialisations, we achieve the same results when we use the convex relaxed Chan-Vese model.

(a) Initialisation 1. (b) Segmentation 1. (c) Initialisation 2. (d) Segmentation 2.

Figure 2.17: Convex relaxed Chan-Vese segmentation results for two different initialisations.

2.2.7. Selective Image Segmentation

In contrast to global image segmentation, in which we segment all objects in an image, selective segmentation aims to segment only a subset of the objects in the image. This generally requires some input from the user to indicate which object or objects they want segmented. Typically, this is in the form of marker points on the image near the object or objects. We denote this set of k marker points by $\mathcal{M}$, i.e.

$$\mathcal{M} = \{\boldsymbol{x}_i \in \Omega, 1 \leq i \leq k\}$$

where $\Omega \subset \mathbb{R}^d$. This marker set $\mathcal{M}$ can be built into the model by, for example, giving an initialisation for the segmentation, or giving location information for where the objects are in the image.

We will initially begin the discussion with the seminal Geodesic Active Contours model of Caselles et al. [36] and the model of Gout et al. [78]. We then build to more sophisticated models which incorporate intensity, distance and area restrictions.

2.2.7.1 Geodesic Active Contours Model (1997)

In 1997, Caselles, Kimmel and Sapiro [36] proposed the Geodesic Active Contours (GAC) model

$$\Gamma^* = \arg\min_{\Gamma} \mathcal{F}_{GAC}(\Gamma)$$

where]]

$$\mathcal{F}_{GAC}(\Gamma) = \int_{\Gamma} g(|\nabla z|)\ \mathrm{d}\Gamma.$$

for a given image z and $g(|\nabla z|)$ the edge detector given by (2.3). In the level set framework, this model can be rewritten as

$$\phi^* = \arg\min_{\phi} \mathcal{F}_{GAC_{LS}}(\phi)$$

where

$$\mathcal{F}_{GAC_{LS}}(\phi) = \int_{\Omega} g(|\nabla z|)|\nabla H_{\varepsilon}(\phi)|\ \mathrm{d}\Omega.$$

which has the Euler-Lagrange equation

$$\delta_{\varepsilon}(\phi)\nabla \cdot \left(g(|\nabla z|)\frac{\nabla \phi}{|\nabla \phi|_{\varepsilon_1}} \right) = 0, \tag{2.19}$$

with Neumann boundary conditions $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ for $\boldsymbol{n}$ the unit outward normal. As before, we set $|\nabla \phi|_{\varepsilon_1} = \sqrt{\phi_x^2 + \phi_y^2 + \varepsilon_1}$, for small $\varepsilon_1 > 0$, to avoid a zero denominator when $|\nabla \phi| = 0$. We may also replace $\delta_{\varepsilon}(\phi)$ by $|\nabla \phi|_{\varepsilon_1} \neq 0$ to ensure the motion is applied to all level sets (rather than just the zero level set), obtaining

$$|\nabla \phi|_{\varepsilon_1} \left(\nabla \cdot \left(g(|\nabla z|)\frac{\nabla \phi}{|\nabla \phi|_{\varepsilon_1}} \right) \right) = 0.$$

This rescaling also makes the flow independent of the scaling of ϕ [5, 178]. It is proven in [36] that this final PDE has a unique (viscosity) solution, i.e. the segmentation result exists and is unique.

2.2.7.2 Gout et al. Model (2005)

The model of Gout et al. [78] builds on the GAC model and incorporates the user-specified marker set $\mathcal{M}$ into a distance function $\mathcal{D}(\boldsymbol{x})$. The model is given by

$$\phi^* = \arg\min_{\phi} \mathcal{F}_{GOUT}(\phi)$$

where

$$\mathcal{F}_{GOUT}(\phi) = \int_\Omega g(|\nabla z|)\mathcal{D}(\boldsymbol{x})|\nabla H_\varepsilon(\phi)| \, \mathrm{d}\Omega.$$

for a given image z. We have some flexibility in the choice of the distance function, the following are two examples used in the literature.

Distance Function 1. We first consider the simple Euclidean distance, defined by

$$\mathcal{D}_1(\boldsymbol{x}) = \min\{|\boldsymbol{x} - \boldsymbol{x}_i|\}, \text{ for all } \boldsymbol{x}_i \in \mathcal{M},$$

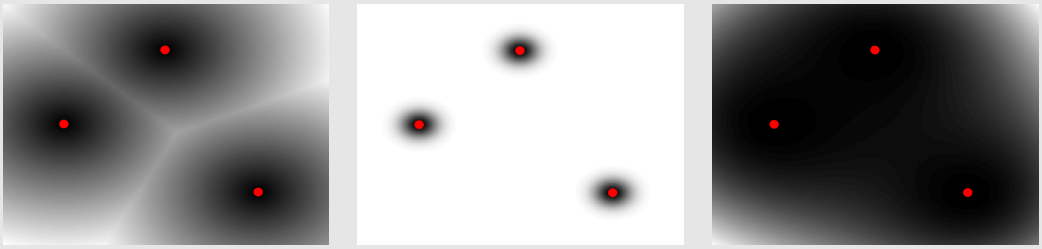
this gives a distance function as in Figure 2.18(a).

Distance Function 2. Secondly, we consider the distance function from [78]

$$\mathcal{D}_2(\boldsymbol{x}) = \prod_{i=1}^{k} \left(1 - \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{x}_i|^2}{2\sigma^2}\right)\right)$$

where σ is a non-negative tuning parameter. The value of σ has a large impact on the resulting distance map. It can be set manually and tuned to the image being segmented, or automatically, e.g. $\sigma = \min_{i,j,i\neq j} |\boldsymbol{x}_i - \boldsymbol{x}_j|$.

Example 2.2.7.1 (Comparison of distance functions). We compare $\mathcal{D}_1(\boldsymbol{x})$ and $\mathcal{D}_2(\boldsymbol{x})$. The set $\mathcal{M}$ is given by the red marker points.



(a) $\mathcal{D}_1$ (b) $\mathcal{D}_2$ with $\sigma = 0.04$ (c) $\mathcal{D}_2$ with automatic σ.

Figure 2.18: Comparing distance functions.

2.2.7.3 Badshah et al. Model (2010)

Badshah and Chen [12] then combined the Gout et al. model [78] with the Chan-Vese model [46] to incorporate a constraint on the intensity in the segmented region, thereby encouraging the contour to segment homogeneous regions. The model is given by

$$(\phi^*, c_1^*, c_2^*) = \arg\min_{\phi} \mathcal{F}_{BC}(\phi, c_1, c_2),$$

where

$$\begin{aligned}\mathcal{F}_{BC}(\phi, c_1, c_2) = \mu \int_\Omega g(|\nabla z|)\mathcal{D}(\boldsymbol{x})|\nabla H_\varepsilon(\phi)| \, \mathrm{d}\Omega \\ + \lambda_1 \int_\Omega (z - c_1)^2 H_\varepsilon(\phi) \, \mathrm{d}\Omega + \lambda_2 \int_\Omega (z - c_2)^2 \left(1 - H_\varepsilon(\phi)\right) \, \mathrm{d}\Omega\end{aligned} \tag{2.20}$$

for z the given image and c_1, c_2 the average intensities of the regions in which ϕ is positive and negative respectively. μ, λ_1 and λ_2 are fixed non-negative real parameters.

2.2.7.4 Nguyen et al. Model (2012)

Nguyen et al. [121] propose a convex model which builds on the GAC model, incorporating intensity fitting terms. The user inputs a marker set $\mathcal{M}$ indicating foreground pixels and an anti-marker set $\mathcal{A}$ which indicates the background pixels. The intensity fitting terms use the sets $\mathcal{M}$ and $\mathcal{A}$ and assign a probability to every pixel in the image for whether it is in the foreground or background. Their model is given by

$$u^* = \arg\min_{u \in [0,1]} \mathcal{F}_{NG}(u)$$

where

$$\mathcal{F}_{NG}(u) = \mu \int_\Omega g(|\nabla z|)|\nabla u| \, \mathrm{d}\Omega + \lambda \int_\Omega r(\boldsymbol{x}) u \, \mathrm{d}\Omega. \tag{2.21}$$

for a given image z with fixed non-negative real parameters μ and λ. Also,

$$r(\boldsymbol{x}) = \alpha \left(P_B(\boldsymbol{x}) - P_F(\boldsymbol{x})\right) + (1 - \alpha)\left(1 - 2P(\boldsymbol{x})\right),$$

where $P_F(\boldsymbol{x})$ and $P_B(\boldsymbol{x})$ are the normalised log likelihood that pixel $\boldsymbol{x}$ belongs to the foreground or background. They are respectively defined by

$$P_F(\boldsymbol{x}) = \frac{-\log Pr(\boldsymbol{x}|F)}{-\log Pr(\boldsymbol{x}|F) - \log Pr(\boldsymbol{x}|B)}$$

and

$$P_B(\boldsymbol{x}) = \frac{-\log Pr(\boldsymbol{x}|B)}{-\log Pr(\boldsymbol{x}|F) - \log Pr(\boldsymbol{x}|B)}.$$

Finally, $P(\boldsymbol{x})$ is the probability that pixel $\boldsymbol{x}$ belongs to the foreground. The first term in $r(\boldsymbol{x})$ ensures the segmentation evolves in accordance with whether a pixel is more likely to be foreground or background, i.e. if $P_B(\boldsymbol{x}) > P_F(\boldsymbol{x})$ then $r(\boldsymbol{x}) > 0$ and $u(\boldsymbol{x})$ tends to decrease. So, at convergence, $u(\boldsymbol{x})$ takes small values near 0 for the background pixels and a value near 1 for the foreground pixels. Also, the second term in $r(\boldsymbol{x})$ ensures the segmentation doesn't drift too far from the initial contour [121]. $\alpha \in [0, 1]$ is a trade-off parameter between the first and second terms.

This model is good for many examples, see [121], however, fails when the boundary of the object to be segmented is non-smooth or has fine structures. Also, the final result is sometimes sensitive to the marker sets used (see Figure 2.19).

Example 2.2.7.2 (Nguyen et al. segmentation results)**.** Here we give some examples of segmentation results from the Nguyen et al. model. The top row shows successful results, the second gives failed results and the final row shows how, for two similar marker sets, we obtain very different results. The first three examples are taken from [121], with the results all obtained using their publicly available software (see [121]).

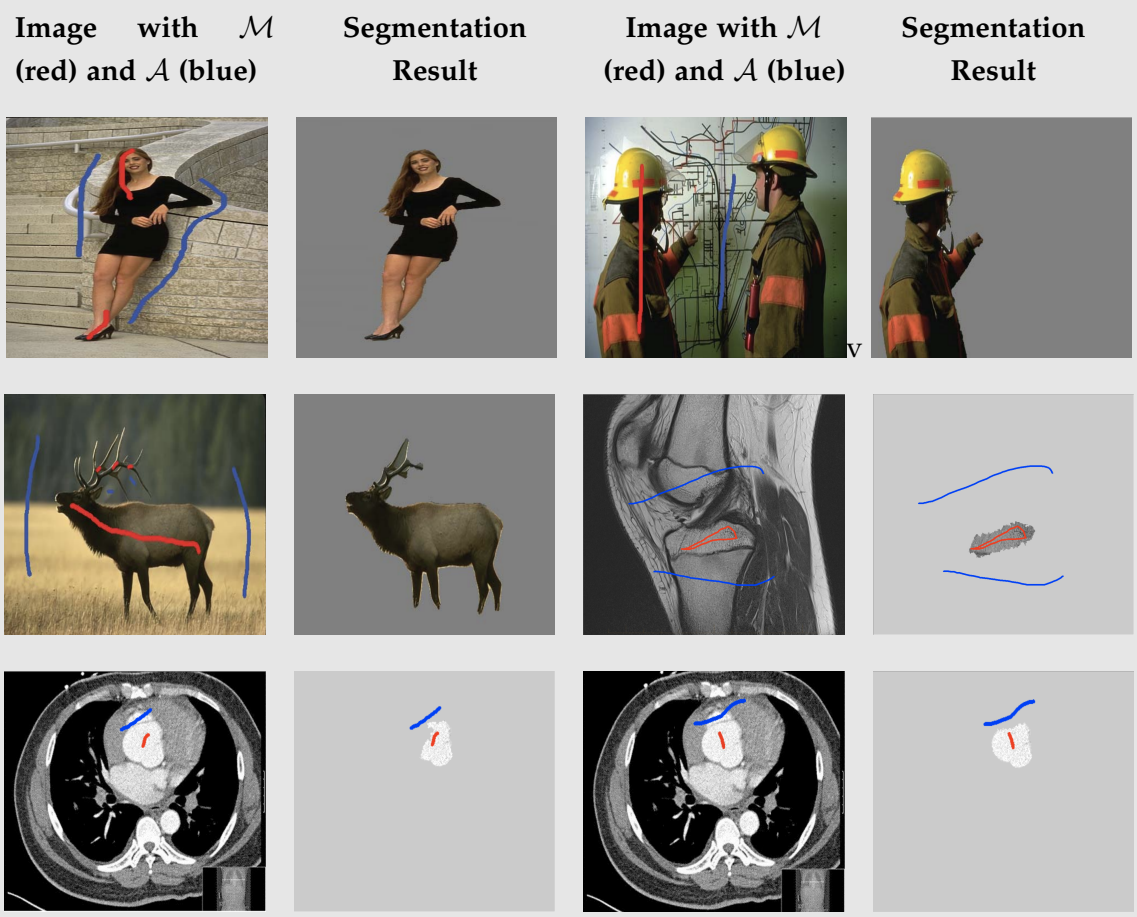


Figure 2.19: Example Nguyen et al. segmentation results.

2.2.7.5 Rada et al. Model (2013)

Of all the models introduced so far, none have a restriction on the size of the object or objects to be segmented. The model of Rada and Chen [135] enforces a penalty in the functional which applies if the area of the segmented object differs from the expected area for the object. Their proposed model is

$$(\phi^*, c_1^*, c_2^*) = \arg\min_{\phi, c_1, c_2} \mathcal{F}_{RC}(\phi, c_1, c_2)$$

where

$$\begin{aligned}\mathcal{F}_{RC}(\phi, c_1, c_2) = \mu \int_\Omega g(|\nabla z|)\mathcal{D}(\boldsymbol{x})|\nabla H_\varepsilon(\phi)| \, \mathrm{d}\Omega \\ + \lambda_1 \int_\Omega (z - c_1)^2 H_\varepsilon(\phi) \, \mathrm{d}\Omega + \lambda_2 \int_\Omega (z - c_2)^2 \left(1 - H_\varepsilon(\phi)\right) \, \mathrm{d}\Omega \\ + \gamma \left[\left(\int_\Omega H_\varepsilon(\phi) \, \mathrm{d}\Omega - A_1 \right)^2 + \left(\int_\Omega (1 - H_\varepsilon(\phi)) \, \mathrm{d}\Omega - A_2 \right)^2 \right].\end{aligned} \tag{2.22}$$

for given image z and fixed non-negative real parameters $\mu, \lambda_1, \lambda_2$ and γ. In this case, A_1 is the area of the object we would like to segment and A_2 is the complementary area, i.e. $A_2 = |\Omega| - A_1 = 1 - A_1$. It is not noted in the original paper, but we can rewrite the final term here more compactly by recognising that

$$\begin{aligned}\left(\int_\Omega (1 - H_\varepsilon(\phi)) \mathrm{d}\Omega - A_2 \right)^2 &= \left(1 - \int_\Omega H_\varepsilon(\phi) \, \mathrm{d}\Omega - 1 + A_1 \right)^2 \\ &= \left(\int_\Omega H_\varepsilon(\phi) \, \mathrm{d}\Omega - A_1 \right)^2\end{aligned} \tag{2.23}$$

so the functional simplifies to

$$\begin{aligned}\mathcal{F}_{RC}(\phi, c_1, c_2) = \mu \int_\Omega g(|\nabla z|)\mathcal{D}(\boldsymbol{x})|\nabla H_\varepsilon(\phi)| \, \mathrm{d}\Omega \\ + \lambda_1 \int_\Omega (z - c_1)^2 H_\varepsilon(\phi) \, \mathrm{d}\Omega + \lambda_2 \int_\Omega (z - c_2)^2 \left(1 - H_\varepsilon(\phi)\right) \, \mathrm{d}\Omega \\ + 2\gamma \left(\int_\Omega H_\varepsilon(\phi) \, \mathrm{d}\Omega - A_1 \right)^2.\end{aligned} \tag{2.24}$$

We can approximate A_1 quite accurately. If we require that the marker set $\mathcal{M}$ is along the boundary of the object to be segmented and form a polygon $\mathcal{P}$ from the points of $\mathcal{M}$, then A_1 is approximated by the area of $\mathcal{P}$. See Figure 2.20(a,b).

2.2.7.6 Klodt et al. Models (2013)

The framework of Klodt et al. [98] builds on the previous constrained minimisation models by incorporating additional constraints on the moments of the solution to the functional. Therefore, for models of the form

$$u^* = \arg\min_{u\in[0,1]} \mathcal{F}(u)$$

and $\mathcal{F}(u)$ an arbitrary convex functional, the Klodt et al. models are generated by simply adding additional convex constraint terms to $\mathcal{F}(u)$. We detail below the first two moment constraints they consider.

Zero-th moment. This is the area of the final segmentation result and is calculated as

$$\int_\Omega u \, \mathrm{d}\Omega.$$

Therefore, if we know the area A, of the object we want to segment, we can include the convex constraint

$$\left(\int_\Omega u \, \mathrm{d}\Omega - A\right)^2$$

in the functional. This constraint is strikingly similar to the area constraint in the Rada et al. model. The key difference is that the term in the Rada et al. model is non-convex, whereas this is convex.

First moment. This is the centroid of the segmentation result, determined by

$$\frac{\int_\Omega \boldsymbol{x} u \, \mathrm{d}\Omega}{\int_\Omega u \, \mathrm{d}\Omega.}.$$

As before, if we know the location of the centroid $\boldsymbol{C}$, we can include the convex constraint

$$\left(\frac{\int_\Omega \boldsymbol{x} u \, \mathrm{d}\Omega}{\int_\Omega u \, \mathrm{d}\Omega.} - \boldsymbol{C} \right)^2 .$$

in the functional.

In fact, we can approximate A and $\boldsymbol{C}$ quite accurately if we demand that the marker set $\mathcal{M}$ is along the boundary of the object to be segmented. If we form a polygon $\mathcal{P}$ from the points of $\mathcal{M}$, then A is approximated by the area of $\mathcal{P}$ and $\boldsymbol{C}$ is its centroid.

Example 2.2.7.3 (Approximating A and $\boldsymbol{C}$ from $\mathcal{M}$)**.** Suppose we have the marker set $\mathcal{M}$ below in (a), the individual markers are shown in red. Then we can form a polygon from these points, shown in (b). Using this polygon, we can approximate the area of the kidney A and also the centroid of the kidney $\boldsymbol{C}$, as shown in (c).

(a) $\mathcal{M}$ (b) $\mathcal{P}$ (c) Approximation to $\boldsymbol{C}$, obtained from $\mathcal{P}$.

Figure 2.20: Comparing distance functions.

Higher-order moments. The paper of Klodt et al. [98] considers higher-order moments and the respective convex constraints which can also be included in the functional. We will not discuss any higher-order moments in this thesis but refer the reader to the paper for further details.

It is important to note that the Klodt et al. model is the first we have reviewed which includes a location constraint on the segmentation result separate from the regularisation term.

2.2.7.7 Spencer et al. Model (2015)

Spencer and Chen [153] introduced a model which also incorporated a location constraint on the segmentation result. Although the Klodt et al. [98] centroid constraint is elegant, the Gâteaux derivative for it is not so simple, explicitly it is

$$2\sum_{i=1}^{d}\left(\int_{\Omega}(\boldsymbol{C}_i-\boldsymbol{x}_i)\,u\,\mathrm{d}\Omega\,(\boldsymbol{C}_i-\boldsymbol{x}_i)\right). \tag{2.25}$$

This is not so simple to solve, with the dependence on u inside the integral, so Spencer et al. proposed a model with the aim to simplify this equation. Their model incorporates the normalised Euclidean distance $\mathcal{D}_E(\boldsymbol{x})$ from $\mathcal{P}$ as its location constraint, the proposed model is

$$(\phi^*, c_1^*, c_2^*) = \arg\min_{\phi,c_1,c_2}\mathcal{F}_{SC}(\phi, c_1, c_2)$$

where

$$\begin{aligned}\mathcal{F}_{SC}(\phi, c_1, c_2) = \mu&\int_{\Omega} g(|\nabla z|)|\nabla H_\varepsilon(\phi)|\,\mathrm{d}\Omega\\ &+\lambda_1\int_{\Omega}(z-c_1)^2H_\varepsilon(\phi)\,\mathrm{d}\Omega+\lambda_2\int_{\Omega}(z-c_2)^2\,(1-H_\varepsilon(\phi))\,\mathrm{d}\Omega\\ &+\theta\int_{\Omega}\mathcal{D}_E(\boldsymbol{x})H_\varepsilon(\phi)\,\mathrm{d}\Omega,\end{aligned} \tag{2.26}$$

for given image z and fixed non-negative real parameters $\mu, \lambda_1, \lambda_2$ and θ.

We now note two properties of this model. Firstly, the regulariser of this model differs from the Rada-Chen model (2.22) as the distance function has been separated from the edge detector term and is now a standalone penalty term $\mathcal{D}_E(\boldsymbol{x})$. Secondly, the distance term $\mathcal{D}_E(\boldsymbol{x})$ is fixed based on the marker set $\mathcal{M}$ (and the resulting polygon $\mathcal{P}$) so we need only compute it once. Therefore, the Gâteaux derivative of the final term in the functional is

$$\theta\mathcal{D}_E(\boldsymbol{x})\delta_\varepsilon(\phi),$$

far simpler than (2.25).

Example 2.2.7.4 (Distance penalty $\mathcal{D}_E(\boldsymbol{x})$)**.** Suppose we have the marker set $\mathcal{M}$ below in (a), the individual markers are shown in red. Then we can form a polygon from

these points, shown in (b). We then calculate the normalised Euclidean distance $\mathcal{D}_E(\boldsymbol{x})$ from $\mathcal{P}$.

(a) $\mathcal{M}$ (b) $\mathcal{P}$ (c) $\mathcal{D}_E(\boldsymbol{x})$

Figure 2.21: Computing $\mathcal{D}_E(\boldsymbol{x})$ for a given $\mathcal{P}$.

2.2.7.8 Convex Relaxed Spencer et al. Model (2015)

Spencer et al. also note [153] that their model belongs to the class of models discussed by Chan et al. [44] for which a convex formulation can be found. This allows us to reformulate their proposed non-convex model to a convex model. The convex relaxed Spencer et al. model is

$$(u^*, c_1^*, c_2^*) = \arg\min_{u,c_1,c_2} \mathcal{F}_{CSC}(u, c_1, c_2)$$

where

$$\begin{aligned}\mathcal{F}_{CSC}(u, c_1, c_2) = \mu \int_\Omega g(|\nabla z|)|\nabla u| \mathrm{d}\Omega + \theta \int_\Omega \mathcal{D}_E(\boldsymbol{x}) u \ \mathrm{d}\Omega \\ + \int_\Omega \left[\lambda_1 (z - c_1)^2 - \lambda_2 (z - c_2)^2\right] u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu(u) \ \mathrm{d}\Omega,\end{aligned} \tag{2.27}$$

for given image z and fixed non-negative real parameters $\mu, \lambda_1, \lambda_2$ and θ with $\nu(u)$ is as defined in §2.2.6.3 (for the convex relaxed reformulation of the Chan-Vese model) and

$$\alpha > \frac{1}{2} \left|\left| \lambda_1 (z - c_1)^2 - \lambda_2 (z - c_2)^2 + \theta \mathcal{D}_E(\boldsymbol{x}) \right|\right|_{L^\infty}$$

To be explicit, the minimisation for u, with c_1 and c_2 fixed, is convex and therefore permits arbitrary initialisation to obtain the selective segmentation result.

2.2.7.9 Liu et al. Model (2018)

Recently, a convex model was introduced by Liu et al. [110] which applies a weighting to the data fitting term, this model is given by

$$u^* = \arg\min_u \mathcal{F}_{LIU}(u)$$

where

$$F_{LIU}(u) = \mu \int_\Omega |\nabla u| \, \mathrm{d}\Omega + \mu_2 \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega + \lambda \int_\Omega \omega^2(\boldsymbol{x}) \, |z-u|^2 \, \mathrm{d}\Omega, \tag{2.28}$$

for given image z and fixed non-negative real parameters μ, μ_2 and λ. Also, $\omega(\boldsymbol{x}) = 1 - \mathcal{D}(\boldsymbol{x})g(|\nabla z|)$ and $\mathcal{D}(\boldsymbol{x})$ is a distance function from marker set $\mathcal{M}$ (such as $\mathcal{D}_1(\boldsymbol{x})$ or $\mathcal{D}_2(\boldsymbol{x})$ from §2.2.7.2).

These are all the variational models which we will consider. In Chapters 3 and 4, we propose two new variational models for selective image segmentation which improve on all the previously discussed models an obtain state-of-the-art results. In the next section, we will discuss how we discretise the domain Ω and transfer our continuous image segmentation models into a discrete setting.

2.3. Multigrid

Not only are accurate segmentation results required, but it is also required that the segmentation method is fast. Many imaging applications demand increasingly higher image resolution e.g. an image of size 25000×25000 (or practically 10^8 unknowns) can be common in oncology imaging. In this section we will give an introduction to fast multigrid methods, both linear and non-linear, used to solve PDEs. In particular, we focus on those PDEs arising from variational models.

Practical multigrid methods were first introduced in the 1970s by Brandt [28]. These methods can solve elliptic PDEs discretized on $N = n \times m$ grid points in $\mathcal{O}(N)$ operations. The multigrid methods can solve general elliptic equations with non-constant coefficients with hardly any loss in efficiency, even non-linear equations can be solved with comparable speed. There is no single multigrid algorithm that solves all elliptic

problems. Rather, there is a multigrid technique that provides the framework for solving these problems. To solve your own problem, one must adjust the various components of the algorithm within this framework. See [28, 49, 89, 158, 172] and the many references therein for more details.

2.3.1. Smoothing Effect of Iterative Solvers

In this section, we will consider the iterative schemes introduced in the previous section, namely Jacobi and Gauss-Seidel. As we will show, these iterative schemes have a remarkable property which makes them incredibly fundamental to a multigrid algorithm – they smooth out the high-frequency Fourier components of the algebraic error. This section will build to this main idea by showing:

1. That the algebraic error at each step can be related to the error at the previous step (and even to the initial algebraic error) using the matrix T obtained from the iterative scheme.
2. A discretised operator (which has a difference stencil) has known eigenvalues and eigenvectors.
3. The idea of the Discrete Fourier Transform allows us to express all functions discretised to a grid in an exponential (Fourier) basis.
4. Therefore the algebraic error can be expressed in a Fourier basis and we can use the eigenvalues of the iterative scheme (a discrete operator) to relate the error at step k to the error at step $k-1$.
5. The eigenvalue for the error is a measure of the amplification of the error in the solution (for a single frequency).
6. The smoothing schemes are effective on the high-frequency components of the error.

Remark 2.3.1.1 (Briefly reverting to vertex-centered discretisation)**.** In this section, we will briefly use the theory obtained from analysis of vertex-centered discretisations of Ω, rather than the cell-centered discretisations we have considered elsewhere. This is primarily for the convenience of the author and to allow clarity for

the reader as it allows us to consider grid points $\boldsymbol{x}$ which can belong to Ω^h and also to Ω^{2h}. The results we find in this section all also apply to cell-centered discretisations, however, the notation is clumsy and confusing. In the literature, we only find one paper which considers the smoothing effect for a cell-centered discretisation [118] and we refer the interested reader to this. We revert back to cell-centered discretisations from §2.3.2 onwards.

2.3.1.1 Relating errors at different iterations

To solve a linear system $A\boldsymbol{x} = \boldsymbol{b}$ for $\boldsymbol{x}$ we can use a linear iterative solver, as discussed in §A.3.2.1. Common examples are Jacobi schemes and Gauss-Seidel schemes.

Definition 2.3.1.2 (Algebraic error)**.** Suppose we have an approximation $\tilde{\boldsymbol{x}}$ to $\boldsymbol{x}$, the solution of $A\boldsymbol{x} = \boldsymbol{b}$. The algebraic error in the approximation is defined as $\boldsymbol{e} = \boldsymbol{x} - \tilde{\boldsymbol{x}}$.

We recall that the Jacobi and Gauss-Seidel schemes can be written in matrix form as

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c} \tag{2.29}$$

where the Jacobi scheme has $T := T_J = D^{-1}R$ and $\boldsymbol{c} := \boldsymbol{c}_J = D^{-1}\boldsymbol{b}$ (see §A.3.2.1.1) and the Gauss-Seidel scheme has $T := T_{GS} = -(D+L)^{-1}U$ and $\boldsymbol{c} := \boldsymbol{c}_{GS} = (D+L)^{-1}\boldsymbol{b}$ (see §A.3.2.1.2). The exact solution $\boldsymbol{x}$ of $A\boldsymbol{x} = \boldsymbol{b}$ satisfies

$$\boldsymbol{x} = T\boldsymbol{x} + \boldsymbol{c}. \tag{2.30}$$

If we subtract (2.29) from (2.30), we get

$$\boldsymbol{e}^{(k)} = T\boldsymbol{e}^{(k-1)}$$

and we see that by applying this repeatedly we obtain

$$\boldsymbol{e}^{(k)} = T^k\boldsymbol{e}^{(0)}.$$

So we can relate the error at step k to the error in the initialisation. Notice that

$$\left|\left|\boldsymbol{e}^{(k)}\right|\right| = \left|\left|T^k\boldsymbol{e}^{(0)}\right|\right| \leq \left|\left|T^k\right|\right| \left|\left|\boldsymbol{e}^{(0)}\right|\right| = ||T||^k \left|\left|\boldsymbol{e}^{(0)}\right|\right|$$

and therefore we get that $\left|\left|\boldsymbol{e}^{(k)}\right|\right| \to 0$ as $k \to \infty$ if $||T|| < 1$. However, this condition is merely sufficient, not necessary, for convergence of the iterative solver. Thankfully, we can use the following theorem (repeated from earlier), which gives both a necessary and sufficient condition for convergence of an iterative algorithm.

Theorem 2.3.1.3 (Spectral Radius and Iterative Convergence)**.** For any $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$, the sequence $\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$ defined by

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c},$$

converges to the unique solution $\boldsymbol{x} = T\boldsymbol{x} + \boldsymbol{c}$ if and only if $\rho(T) < 1$.

2.3.1.2 Eigenfunctions for Discrete Operators

Suppose we have a discrete operator $\mathcal{N}^h$ which has a difference stencil $\boldsymbol{s}^h = \{s_{\boldsymbol{k}}\}_{\boldsymbol{k}\in K}$ such that

$$\mathcal{N}^h u^h(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in K} \boldsymbol{s_k} u^h(\boldsymbol{x} + \boldsymbol{kh})$$

where the $\boldsymbol{s_k}$ are constant and K is a finite set.

Example 2.3.1.4 (Laplace operator stencil)**.** The stencil of the discrete 2D Laplace operator $-\Delta^h$ is given intuitively by

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

and the corresponding difference stencil is

$$\boldsymbol{s} = \left\{ s_{(0,-1)} = -\frac{1}{h^2}, s_{(-1,0)} = -\frac{1}{h^2}, s_{(0,0)} = \frac{4}{h^2}, s_{(1,0)} = -\frac{1}{h^2}, s_{(0,1)} = -\frac{1}{h^2} \right\}.$$

We note that for an infinite grid

$$\mathcal{N}^h e^{i\boldsymbol{\theta}\boldsymbol{x}/\boldsymbol{h}} = \sum_{\boldsymbol{k}\in K} \boldsymbol{s}_{\boldsymbol{k}} e^{i\boldsymbol{\theta}(\boldsymbol{x}/\boldsymbol{h}+\boldsymbol{k})} = \left[\sum_{\boldsymbol{k}\in K} \boldsymbol{s}_{\boldsymbol{k}} e^{i\boldsymbol{\theta}\boldsymbol{k}}\right] e^{i\boldsymbol{\theta}\boldsymbol{x}/\boldsymbol{h}} \tag{2.31}$$

and therefore $\varphi^h(\boldsymbol{\theta}, \boldsymbol{x}) = e^{i\boldsymbol{\theta}\boldsymbol{x}/\boldsymbol{h}}$ is an eigenfunction of the discrete operator $\mathcal{N}^h$ with eigenvalue $\mu^h(\boldsymbol{\theta}) = \sum_{\boldsymbol{k}\in K} \boldsymbol{s}_{\boldsymbol{k}} e^{i\boldsymbol{\theta}\boldsymbol{k}}$.

In our case, the grid is not infinite and we must also consider the boundary conditions of our domain. We find that the eigenvalues and eigenvectors are the same if we assume periodic boundary conditions but, ass we will see in Example 2.3.1.5, Dirichlet boundary conditions change the matrix structure of the operator and change the eigenvectors and eigenvalues.

Example 2.3.1.5 (Laplace operator eigenvalues)**.**

1-D Dirichlet Boundary Conditions.

The one-dimensional discretised Laplace operator with Dirichlet boundary conditions, i.e. $u(0) = 0$ and $u(N+1) = 0$ is given by

$$-\Delta^h_{\text{Dirichlet}} = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 & 0 \\ -1 & 2 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 2 & -1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & \ddots & -1 \\ 0 & 0 & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}.$$

Lemma. This tridiagonal matrix has eigenvectors $\boldsymbol{v}_k$ and eigenvalues μ_k where

$$\boldsymbol{v}_k[j] = \sin\left(\frac{\pi jk}{N}\right)$$

for $j, k = 1, \ldots, N$ and eigenvalues

$$\mu_k = \frac{2}{h^2}\left[1 - \cos\left(\frac{\pi k}{N}\right)\right].$$

Proof. We will consider the discrete operator applied to just the component $\boldsymbol{v}_k[j]$

$$\begin{aligned}
-\Delta^h\left(\sin\left(\frac{\pi jk}{N}\right)\right) &= \frac{2\sin\left(\frac{\pi jk}{N}\right) - \sin\left(\frac{\pi(j+1)k}{N}\right) - \sin\left(\frac{\pi(j-1)k}{N}\right)}{h^2} \\
&= \frac{1}{h^2}\left[2\sin\left(\frac{\pi jk}{N}\right) - \sin\left(\frac{\pi jk}{N}\right)\cos\left(\frac{\pi k}{N}\right)\right. \\
&\quad - \sin\left(\frac{\pi k}{N}\right)\cos\left(\frac{\pi jk}{N}\right) - \sin\left(\frac{\pi jk}{N}\right)\cos\left(\frac{\pi k}{N}\right) \\
&\quad \left. + \sin\left(\frac{\pi k}{N}\right)\cos\left(\frac{\pi jk}{N}\right)\right] \\
&= \frac{2}{h^2}\left[1 - \cos\left(\frac{\pi k}{N}\right)\right]\sin\left(\frac{\pi jk}{N}\right)
\end{aligned}$$

therefore $\boldsymbol{v}_k$ are the eigenvectors with corresponding eigenvalues

$$\mu_k = \frac{2}{h^2}\left[1 - \cos\left(\frac{\pi k}{N}\right)\right].$$

Generalisation to d-D with Dirichlet Boundary Conditions. It immediately follows from the previous reasoning that the d-dimensional discretised Laplace operator with Dirichlet boundary conditions has eigenvectors

$$\boldsymbol{v}_{k_1,k_2,\ldots,k_d}[j] = \prod_{i=1}^{d}\sin\left(\frac{\pi jk_i}{n_i}\right)$$

where $k_i = 1, \ldots, n_i$. The corresponding eigenvalues

$$\mu_k = \frac{1}{h^2}\left[2^d - \sum_{i=1}^{d}\cos\left(\frac{\pi k_i}{n_i}\right)\right].$$

1-D Periodic Boundary Conditions.

The one-dimensional discretised Laplace operator with periodic boundary conditions, i.e. $u[i \pm kN] = u[i]$ for $i \in [1, N] \subset \mathbb{N}$ and $k \in \mathbb{Z}$, is given by

$$-\Delta^h_{\text{Periodic}} = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 & -1 \\ -1 & 2 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 2 & -1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & \ddots & -1 \\ -1 & 0 & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}$$

Lemma. This matrix has a circulant structure and we have eigenvectors given by $\boldsymbol{v}_k$ where

$$\boldsymbol{v}_k[j] = e^{\boldsymbol{i}\frac{2\pi k}{N}(x_j/h)} = e^{\boldsymbol{i}\frac{2\pi k}{N}j}.$$

for $j, k = 1, \ldots, N$. The eigenvalues are

$$\mu_k = \frac{2}{h^2}\left[1 - \cos\left(\frac{2\pi k}{N}\right)\right]$$

Proof. We will consider the discrete operator applied to just the component $\boldsymbol{v}_k[j]$

$$\begin{aligned} -\Delta^h e^{\boldsymbol{i}\frac{2\pi k}{N}j} &= \frac{2e^{\boldsymbol{i}\frac{2\pi k}{N}j} - e^{\boldsymbol{i}\frac{2\pi k}{N}(j+1)} - e^{\boldsymbol{i}\frac{2\pi k}{N}(j-1))}}{h^2} \\ &= \frac{1}{h^2}\left[2e^{\boldsymbol{i}\frac{2\pi k}{N}j} - e^{\boldsymbol{i}\frac{2\pi k}{N}}e^{\boldsymbol{i}\frac{2\pi k}{N}j} - e^{-\boldsymbol{i}\frac{2\pi k}{N}}e^{\boldsymbol{i}\frac{2\pi k}{N}j}\right] \\ &= \frac{1}{h^2}\left[2 - e^{\boldsymbol{i}\frac{2\pi k}{N}} - e^{-\boldsymbol{i}\frac{2\pi k}{N}}\right]e^{\boldsymbol{i}\frac{2\pi k}{N}j} \\ &= \frac{2}{h^2}\left[1 - \cos\left(\frac{2\pi k}{N}\right)\right]e^{\boldsymbol{i}\frac{2\pi k}{N}j} \end{aligned}$$

therefore $\boldsymbol{v}_k$ are eigenvectors with corresponding eigenvalue

$$\mu_k = \frac{2}{h^2}\left[1 - \cos\left(\frac{2\pi k}{N}\right)\right].$$

Generalisation to d-D with Periodic Boundary Conditions.

It follows from this that the discretised d-dimensional Laplace operator has eigenvectors given by

$$\boldsymbol{v}(\boldsymbol{k},\boldsymbol{j}) = \prod_{i=1}^{d} e^{\boldsymbol{i}\frac{2\pi k_i j_i}{n_i}} = e^{\boldsymbol{i}2\pi\boldsymbol{k}\cdot\frac{\boldsymbol{j}}{\boldsymbol{n}}}$$

where $\boldsymbol{k} = (k_1, k_2, \ldots, k_d)$ and $\boldsymbol{j} = (j_1, j_2, \ldots, j_d)$ and $\boldsymbol{n} = (n_1, n_2, \ldots, n_d)$ where $N = \prod_{i=1}^{d} n_i$. The eigenvalues are

$$\mu(\boldsymbol{k}) = \frac{1}{h^2}\left[2^d - \sum_{i=1}^{d} \cos\left(\frac{2\pi k_i}{n_i}\right)\right].$$

Therefore, by recognising that the iterative solvers we discussed earlier (Jacobi and Gauss-Seidel) are discrete operators, we can use the result (2.31) to show that for $\varphi^h(\boldsymbol{\theta},\boldsymbol{x}) = e^{\boldsymbol{i}\boldsymbol{\theta}\boldsymbol{x}/\boldsymbol{h}}$, we have

$$T^k\varphi^h(\boldsymbol{\theta},\boldsymbol{x}) = \mu^h(\boldsymbol{\theta})^k\varphi^h(\boldsymbol{\theta},\boldsymbol{x})$$

for each of the iterative schemes discussed earlier (e.g. T_J and T_{GS}). In the next part, we will show, using the Discrete Fourier Transform, that these $\varphi^h(\boldsymbol{\theta},\boldsymbol{x})$ form a basis for the space $\mathbb{C}^N$ and we can write all functions discretised on the domain Ω in terms of these quantities.

2.3.1.3 Discrete Fourier Transform

Suppose that we have a one-dimensional function $u(\boldsymbol{x})$ for which we have values at N points $\{x_j\}_{j=0}^{j=N-1}$. We assume that u is periodic and repeats outside of $[0, N-1]$, i.e. $u(x_N) = u(x_0)$. The Discrete Fourier Transform (DFT) of this data is given by

$$\mathcal{F}\left(u\left(x_j\right)\right) = \sum_{k=0}^{N-1} u\left(x_j\right) e^{-\boldsymbol{i}\frac{2\pi k}{N}j}$$

where $\boldsymbol{i} = \sqrt{-1}$. This is computed for $j = 0, 1, \ldots, N-1$ giving the DFT coefficients. We can write the whole system as

$$\begin{bmatrix} \mathcal{F}\left(u\left(x_0\right)\right) \\ \mathcal{F}\left(u\left(x_1\right)\right) \\ \mathcal{F}\left(u\left(x_2\right)\right) \\ \vdots \\ \mathcal{F}\left(u\left(x_{N-1}\right)\right) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{(N-1)} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)^2} \end{bmatrix} \begin{bmatrix} u\left(x_0\right) \\ u\left(x_1\right) \\ u\left(x_2\right) \\ \vdots \\ u\left(x_{N-1}\right) \end{bmatrix}$$

where $\omega = e^{-\boldsymbol{i}\frac{2\pi}{N}}$. We now have a system $\mathcal{F}(u) = Au$. Suppose that we are interested in the inverse of this, i.e. obtaining the solution u from the Fourier coefficients. We must, therefore, invert A. Thankfully, by noting that $1 + \omega + \cdots + \omega^{N-1} = 0$, we can obtain the inverse by simple computation. It is given by

$$A^{-1} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \cdots & \omega^{-(N-1)} \\ 1 & \omega^{-2} & \omega^{-4} & \cdots & \omega^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)} & \omega^{-2(N-1)} & \cdots & \omega^{-(N-1)^2} \end{bmatrix}.$$

Using this, we can obtain u by $u = A^{-1}\mathcal{F}(u)$ and explicitly we obtain a component $u(x_j)$ by

$$u\left(x_j\right) = \frac{1}{N} \sum_{k=0}^{N-1} \mathcal{F}\left(u\left(x_k\right)\right) e^{\boldsymbol{i}\frac{2\pi k}{N} j} = \sum_{k=0}^{N-1} c_k e^{\boldsymbol{i}\frac{2\pi k}{N} j}$$

where $c_k = \frac{1}{N} \sum_{\ell=0}^{N-1} u\left(x_k\right) e^{-\boldsymbol{i}\frac{2\pi \ell}{N} k}$. Although going through this whole process to arrive back at $u(x_j)$ may seem peculiar, the key aim is to show that we can represent $u(x_j)$ with exponential functions as the basis.

We note also that $e^{-\boldsymbol{i}\frac{2\pi k}{N}(j+N)} = e^{-\boldsymbol{i}2\pi k} e^{-\boldsymbol{i}\frac{2\pi k}{N} j} = e^{-\boldsymbol{i}\frac{2\pi k}{N} j}$. Therefore we can adjust the summation bounds to be symmetric around $k = 0$ (consistent with the multigrid literature), i.e.

$$u\left(x_j\right) = \sum_{k=-m}^{m-p} c_k e^{\boldsymbol{i}\frac{2\pi k}{N} j}$$

where $m = N/2$ and $p = -1$ for N even and $m = (N-1)/2$ and $p = 0$ for N odd. This

can be written as

$$u\left(x_j\right) = \begin{bmatrix} c_{-m} \\ c_{-m+1} \\ \vdots \\ c_{m-p} \end{bmatrix} \cdot \begin{bmatrix} e^{\boldsymbol{i}\frac{2\pi}{N}(-m)j} \\ e^{\boldsymbol{i}\frac{2\pi}{N}(-(m-1))j} \\ \vdots \\ e^{\boldsymbol{i}\frac{2\pi}{N}(m-p)j} \end{bmatrix} = \begin{bmatrix} c_{-m} \\ c_{-m+1} \\ \vdots \\ c_{m-p} \end{bmatrix} \cdot \begin{bmatrix} \varphi^h(\theta_{-m}, x_j) \\ \varphi^h(\theta_{-(m-1)}, x_j) \\ \vdots \\ \varphi^h(\theta_{m-p}, x_j) \end{bmatrix} = \boldsymbol{c} \cdot \boldsymbol{\varphi}_j$$

where

$$\boldsymbol{\varphi}_j = \begin{bmatrix} \varphi^h(\theta_{-m}, x_j) \\ \varphi^h(\theta_{-(m-1)}, x_j) \\ \vdots \\ \varphi^h(\theta_{m-p}, x_j) \end{bmatrix}.$$

We now aim to show that the vectors $\boldsymbol{\varphi}_j$ form a basis. If we can show that $\{\boldsymbol{\varphi}_j\}_{j=-m}^{m-p}$ are mutually linearly independent and orthogonal, then they must span $\mathbb{C}^N$ and be a basis for $\mathbb{C}^N$. The complex inner product of $\boldsymbol{\varphi}_j$ and $\boldsymbol{\varphi}_\ell$ is

$$\begin{aligned} \langle \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_\ell \rangle &= \boldsymbol{\varphi}_j^T \overline{\boldsymbol{\varphi}}_\ell \\ &= e^{\boldsymbol{i}\frac{2\pi}{N}(-m)j} e^{-\boldsymbol{i}\frac{2\pi}{N}(-m)\ell} + e^{\boldsymbol{i}\frac{2\pi}{N}(-(m-1))j} e^{-\boldsymbol{i}\frac{2\pi}{N}(-(m-1))\ell} + \cdots + e^{\boldsymbol{i}\frac{2\pi}{N}(m-p)j} e^{-\boldsymbol{i}\frac{2\pi}{N}(m-p)\ell} \\ &= \sum_{k=-m}^{m-p} e^{\boldsymbol{i}\frac{2\pi}{N}k(j-\ell)} \end{aligned}$$

Now, if $j = \ell$ then we have $\langle \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_\ell \rangle = N$ as each term in the sum is 1. If $j \neq \ell$ we have

$$\langle \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_\ell \rangle = \sum_{k=-m}^{m-p} \left(e^{\boldsymbol{i}\frac{2\pi}{N}(j-\ell)}\right)^k = \frac{1 - e^{\left(\boldsymbol{i}\frac{2\pi}{N}(j-\ell)\right)N}}{1 - e^{\boldsymbol{i}\frac{2\pi}{N}(j-\ell)}} = \frac{1 - e^{(\boldsymbol{i}2\pi(j-\ell))}}{1 - e^{\boldsymbol{i}\frac{2\pi}{N}(j-\ell)}} = 0$$

as $j - \ell \in \mathbb{Z}$. Therefore the N vectors $\{\boldsymbol{\varphi}_j\}_{j=-m}^{m-p}$ are linearly independent and orthogonal. Hence they form a basis for $\mathbb{C}^N$ and we can write any function in the form $u(\boldsymbol{x}) = \sum_{k=-m}^{m-p} c_k \boldsymbol{\varphi}_k$. This idea extends naturally into higher dimensions and all grid functions in d-dimensions can be written in the form

$$u(\boldsymbol{x}) = \sum_{\boldsymbol{k}=-\boldsymbol{m}}^{\boldsymbol{m}-\boldsymbol{p}} \boldsymbol{c_k} \boldsymbol{\varphi_k}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_d), \boldsymbol{m} = (m_1, m_2, \ldots, m_d), \boldsymbol{p} = (p_1, p_2, \ldots, p_d)$ and $\boldsymbol{k} = (k_1, k_2, \ldots, k_d)$. At this point we again change notation by noting $\theta_k := \frac{2\pi k}{N} \in [-\pi, \pi)$ for $k = -m, \ldots, m - p$ and we define the set of frequencies $\Theta = \{\theta_k\}_{k=-m}^{m-p}$. Therefore, the algebraic error can be expressed

$$\boldsymbol{e}(\boldsymbol{x}) = \sum_{\theta \in \Theta} \boldsymbol{d}_\theta \boldsymbol{\varphi}_\theta.$$

In particular, we can express the initial error as

$$\boldsymbol{e}^{(0)}(\boldsymbol{x}) = \sum_{\theta \in \Theta} \boldsymbol{d}_\theta^{(0)} \boldsymbol{\varphi}_\theta$$

and we also have $\boldsymbol{e}^{(k)} = T^k \boldsymbol{e}^{(0)}$. We know that $\boldsymbol{\varphi}_\theta$ are eigenvectors for T (by §2.3.1.2) and therefore

$$T\boldsymbol{\varphi}_\theta = \mu(\theta)\boldsymbol{\varphi}_\theta$$

and with k smoothing steps we have

$$T^k \boldsymbol{\varphi}_\theta = \mu^k(\theta)\boldsymbol{\varphi}_\theta.$$

We suppose $\boldsymbol{d}_\theta$ is fixed and observe that

$$\begin{aligned} \boldsymbol{e}^{(1)}(\boldsymbol{x}) &= \sum_{\theta \in \Theta} \boldsymbol{d}_\theta^{(1)} \boldsymbol{\varphi}_\theta \\ &= T\left(\boldsymbol{e}^{(0)}(\boldsymbol{x})\right) = T\left(\sum_{\theta \in \Theta} \boldsymbol{d}_\theta^{(0)} \boldsymbol{\varphi}_\theta\right) = \sum_{\theta \in \Theta} \boldsymbol{d}_\theta^{(0)} T\left(\boldsymbol{\varphi}_\theta\right) = \sum_{\theta \in \Theta} \boldsymbol{d}_\theta^{(0)} \mu(\theta) \boldsymbol{\varphi}_\theta. \end{aligned}$$

Suppose we split the linear operator $\mathcal{L}$ to $\mathcal{L} = \mathcal{L}^+ + \mathcal{L}^-$ where

$$\mathcal{L}u^{(k)}(\boldsymbol{x}) = \mathcal{L}^+ u^{(k+1)}(\boldsymbol{x}) + \mathcal{L}^- u^{(k)}(\boldsymbol{x}).$$

Then by subtracting this from the exact solution $\mathcal{L}u = \mathcal{L}^+ u + \mathcal{L}^- u$ we obtain the relation

$$\mathcal{L}^+ \boldsymbol{e}^{(k+1)}(\boldsymbol{x}) + \mathcal{L}^- \boldsymbol{e}^{(k)}(\boldsymbol{x}) = \boldsymbol{0}.$$

Let us focus just on one eigenvector of the error, for frequency $\boldsymbol{\theta}$ given by $\varphi(\boldsymbol{\theta}, \boldsymbol{x}) = e^{\boldsymbol{i\theta x/h}}$, therefore

$$\mathcal{L}^+ \varphi^{(k+1)}(\boldsymbol{\theta}, \boldsymbol{x}) + \mathcal{L}^- \varphi^{(k)}(\boldsymbol{\theta}, \boldsymbol{x}) = \boldsymbol{0}.$$

$$\Rightarrow \mathcal{L}^+ T\left(\varphi^{(k)}(\boldsymbol{\theta},\boldsymbol{x})\right) + \mathcal{L}^- \varphi^{(k)}(\boldsymbol{\theta},\boldsymbol{x}) = \mathbf{0}.$$

therefore, if we define $\tilde{\mathcal{L}}^+$ and $\tilde{\mathcal{L}}^-$ as the eigenvalues for the operators $\mathcal{L}^+$ and $\mathcal{L}^-$ respectively, we obtain

$$\tilde{\mathcal{L}}^+ \mu(\boldsymbol{\theta}) \varphi^{(k)}(\boldsymbol{\theta},\boldsymbol{x}) + \tilde{\mathcal{L}}^- \varphi^{(k)}(\boldsymbol{\theta},\boldsymbol{x}) = \mathbf{0},$$

and find the eigenvalue of the smoothing scheme $\mu(\boldsymbol{\theta})$ as

$$\mu(\boldsymbol{\theta}) = -\frac{\tilde{\mathcal{L}}^-(\boldsymbol{\theta})}{\tilde{\mathcal{L}}^+(\boldsymbol{\theta})}.$$

This eigenvalue $\mu(\boldsymbol{\theta})$ is a measure of the degree to which the component $\varphi(\boldsymbol{\theta},\boldsymbol{x})$ is amplified or dampened between iterations. As we are interested only in the absolute size of the amplification (or damping) of error components, we will, in fact, use the following definition for the amplification factor

$$\mu(\boldsymbol{\theta}) := \left|\frac{\tilde{\mathcal{L}}^-(\boldsymbol{\theta})}{\tilde{\mathcal{L}}^+(\boldsymbol{\theta})}\right|$$

Example 2.3.1.6 (Amplification factor for the Helmholtz equation)**.** The inhomogeneous Helmholtz equation is given by

$$-\Delta u + \gamma u = f$$

which is discretised in 1D using standard finite differences as

$$-\frac{u_{i-1}}{h^2} + \left(\frac{2}{h^2} + \gamma\right) u_i - \frac{u_{i+1}}{h^2} = f_i.$$

The Gauss-Seidel scheme is given by

$$-\frac{u_{i-1}^{(k+1)}}{h^2} + \left(\frac{2}{h^2} + \gamma\right) u_i^{(k+1)} - \frac{u_{i+1}^{(k)}}{h^2} = f_i^{(k)}$$

which has error form

$$-e_{i-1}^{(k+1)} + \left(2+\gamma h^2\right) e_i^{(k+1)} - e_{i+1}^{(k)} = 0.$$

In the notation of $\mathcal{L}^+$ and $\mathcal{L}^-$ we can rewrite this as

$$\mathcal{L}^+(\theta)e^{i\theta x_i/h} = -e^{i\theta(x_i/h-1)} + \left(2+\gamma h^2\right) e^{i\theta x_i/h} = \left(\left(2+\gamma h^2\right) - e^{-i\theta}\right) e^{i\theta x_i/h}$$

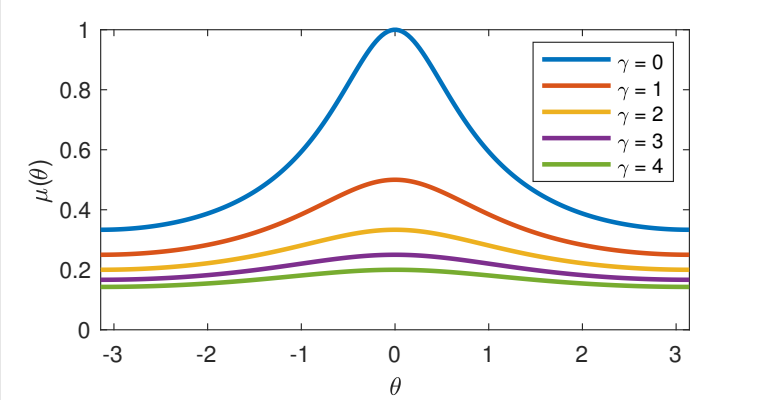
and

$$\mathcal{L}^-(\theta)e^{i\theta x_i/h} = -e^{i\theta(x_i/h+1)} = -e^{i\theta}e^{i\theta x_i/h}$$

Therefore the amplification factor for frequency θ is

$$\mu(\theta) = \left|\frac{e^{i\theta}}{(2+\gamma h^2) - e^{-i\theta}}\right|.$$

We plot $\mu(\theta)$ for several values of γ below.



2.3.1.4 High and Low Frequencies

If we consider the graph in the previous example, we see that consistently the amplification factor is larger in the range $\Theta_L = \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$ than in the range $\Theta_H = [-\pi, \pi) \setminus \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$. Crucially, we can see that the Gauss-Seidel iterative scheme has a small eigenvalue in the range Θ_H and therefore reduces the error component with frequencies in Θ_H quite effectively. This result is also true for lexicographic Jacobi and SOR iterative schemes and also for line and plane Jacobi, Gauss-Seidel and SOR iterative schemes [158].

Definition 2.3.1.7 (High and Low Frequencies)**.** The set of frequencies given by $\Theta_H = [-\pi, \pi) \setminus \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$ is called the set of high frequencies, and $\Theta_L = \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$ is the set of low frequencies. An eigenvector of the error, given by $\varphi(\theta_k, x_j) = e^{\boldsymbol{i}\theta_k x_j/h}$, is called a high-frequency component of the error if $\theta_k \in \Theta_H$ and is called a low-frequency component if $\theta_k \in \Theta_L$.

We can prove algebraically that for the inhomogeneous Helmholtz equation in the previous example, the derivative of $\mu(\theta)$ is positive in $(-\pi, 0)$ and negative in $(0, \pi)$ for all values of γ. We also have

$$\mu(-\pi) = \frac{1}{\gamma+3} \qquad \mu(-\pi/2) = \frac{1}{\sqrt{(\gamma+2)^2+1}} \qquad \mu(0) = \frac{1}{\gamma+1}$$

$$\mu(\pi/2) = \frac{1}{\sqrt{(\gamma+2)^2+1}} \qquad \mu(\pi) = \frac{1}{\gamma+3}$$

and therefore

$$\max_{\theta\in\Theta_H} \mu(\theta) = \frac{1}{\sqrt{(\gamma+2)^2+1}} \qquad \max_{\theta\in\Theta_L} \mu(\theta) = \frac{1}{\gamma+1}$$

Definition 2.3.1.8 (Smoothing rate)**.** The smoothing rate for an iterative scheme is defined by

$$\hat{\mu} = \max_{\theta\in\Theta_H} \mu(\theta)$$

2.3.1.5 Smoothing the Low Frequencies

Let us use the notation from previous sections and proceed in 2D, denoting by $\Omega^h \subset [0,1]^2$ the $n \times n$ discretised grid with spacing h. We follow this with $\Omega^{2h} \subset [0,1]^2$ the $\frac{n}{2} \times \frac{m}{2}$ discretised grid with spacing $2h$ and $\Omega^{4h}, \Omega^{8h}, \ldots$ are defined in a similar fashion.

Let us consider the error component $\varphi^h(\theta_k, x_j) = e^{\boldsymbol{i}\theta_k x_j/h}$ where $x_j \in \Omega^h$ and $x_j \in \Omega^{2h}$.

We introduce the h in φ to be clear which grid we are considering for the x_j. Then

$$\varphi^h(\theta_k, x_j) = e^{\boldsymbol{i}\theta_k x_j/h} = e^{\boldsymbol{i}2\theta_k x_j/2h} = \varphi^{2h}(2\theta_k, x_j).$$

In this case, suppose that θ_k is in the low-frequency set $\Theta_L = \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$ on Ω^h, then we find $2\theta_k \in [-\pi, \pi]$ is potentially in the high-frequency set $\Theta_H = [-\pi, \pi) \setminus \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$ on Ω^{2h}. The key idea here is that by considering the low-frequency error components from Ω^h on the coarser grid Ω^{2h} we can now dampen some of them by using the smoothing iterative schemes such as Jacobi, Gauss-Seidel or SOR. This procedure can be continued through $\Omega^{4h}, \Omega^{8h}, \ldots$, each time we eliminate more and more of the low-frequency error components. Finally, we choose a coarse grid $\Omega^{2^\ell h}$ with $\frac{n}{2^\ell} \times \frac{n}{2^\ell}$ pixels on which we solve the equation exactly using a highly accurate algorithm which can reduce the remaining low-frequency error components. This algorithm can be very computationally expensive however, as we are only computing on a small fraction of the original pixels, namely $\frac{n}{2^\ell} \times \frac{n}{2^\ell}$ of them. The smoothing schemes are cheap to implement compared to the exact solver and therefore if the smoothing schemes dampen the high-frequency errors effectively, multigrid schemes can achieve rapid convergence in an optimal $\mathcal{O}(N)$ operations.

The key reliance here is on the effectiveness of the smoother. If this does not dampen the high-frequency errors effectively, we have the problem where error components on Ω^h will alias with another on Ω^{2h} and we do not achieve convergence. We will discuss aliasing in the next section.

2.3.1.6 Aliasing of Components

Let us consider the 2D error component on Ω^h given by

$$\varphi^h(\boldsymbol{\theta}, \boldsymbol{x}) = e^{\boldsymbol{i}\boldsymbol{\theta}\boldsymbol{x}/h} = e^{\boldsymbol{i}\theta_1 x_1/h} e^{\boldsymbol{i}\theta_2 x_2/h}$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2)$ and $\boldsymbol{x} = (x_1, x_2)$. Let us assume $\boldsymbol{x} \in \Omega^h$ and $\boldsymbol{x} \in \Omega^{2h}$, then we have the following relations

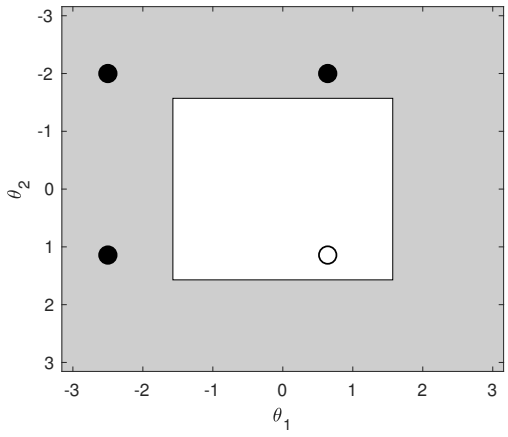
$$\varphi^h(\boldsymbol{\theta}, \boldsymbol{x}) = e^{\boldsymbol{i}\frac{2\theta_1 x_1}{2h}} e^{\boldsymbol{i}\frac{2\theta_2 x_2}{2h}} = \varphi^{2h}(2\boldsymbol{\theta}, \boldsymbol{x})$$

$$\varphi^h\left(\boldsymbol{\theta} \pm (\pi, 0), \boldsymbol{x}\right) = e^{\boldsymbol{i}\frac{2(\theta_1 \pm \pi)x_1}{2h}} e^{\boldsymbol{i}\frac{2\theta_2 x_2}{2h}} = e^{\boldsymbol{i}\frac{\pm 2\pi x_1}{2h}} e^{\boldsymbol{i}\frac{2\theta_1 x_1}{2h}} e^{\boldsymbol{i}\frac{2\theta_2 x_2}{2h}} = \varphi^{2h}(2\boldsymbol{\theta}, \boldsymbol{x})$$

$$\varphi^h\left(\boldsymbol{\theta} \pm (0,\pi), \boldsymbol{x}\right) = e^{\boldsymbol{i}\frac{2\theta_1 x_1}{2h}} e^{\boldsymbol{i}\frac{2(\theta_2 \pm \pi)x_2}{2h}} = e^{\boldsymbol{i}\frac{2\theta_1 x_1}{2h}} e^{\boldsymbol{i}\frac{\pm 2\pi x_2}{2h}} e^{\boldsymbol{i}\frac{2\theta_2 x_2}{2h}} = \varphi^{2h}(2\boldsymbol{\theta}, \boldsymbol{x})$$

$$\varphi^h\left(\boldsymbol{\theta} + (\pm\pi, \pm\pi), \boldsymbol{x}\right) = e^{\boldsymbol{i}\frac{2(\theta_1 \pm \pi)x_1}{2h}} e^{\boldsymbol{i}\frac{2(\theta_2 \pm \pi)x_2}{2h}} = e^{\boldsymbol{i}\frac{\pm 2\pi x_1}{2h}} e^{\boldsymbol{i}\frac{2\theta_1 x_1}{2h}} e^{\boldsymbol{i}\frac{\pm 2\pi x_2}{2h}} e^{\boldsymbol{i}\frac{2\theta_2 x_2}{2h}} = \varphi^{2h}(2\boldsymbol{\theta}, \boldsymbol{x})$$

Therefore, for any given $\boldsymbol{\theta} \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)^2$ we have one error component on Ω^{2h} and four possible components on Ω^h which all appear the same as it on Ω^{2h}. In the image below, we plot an example of a set of frequencies for which all $\varphi^h(\boldsymbol{\theta}, \boldsymbol{x})$ appear the same on Ω^{2h}. The grey area represents the high-frequency set Θ_H on Ω^h and the white area is the low-frequency set Θ_L on Ω^h. The black dots are frequencies on Ω^h and these three components all appear the same as the black circle on Ω^{2h}. Therefore, if the high-frequency errors are not damped on Ω^h then we run into problems.

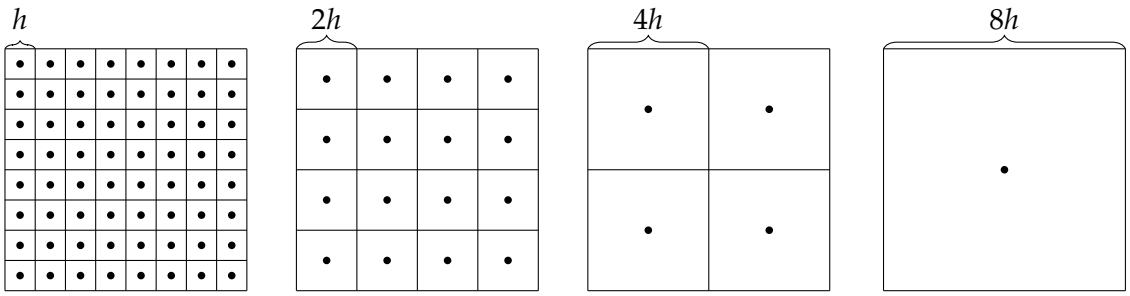


Now that we have introduced the importance of the smoothing operator and explained that it damps out high-frequency error components, we will introduce the other elements of multigrid. These are the transfer operators, which allow us to move between Ω^h and Ω^{2h}, the exact solver on the coarsest grid and the actual architecture of the algorithm we use for the multigrid solver within the multigrid framework.

2.3.2. Restriction and Interpolation Operators

In this section, we will discuss the operators we use to transfer quantities between different grid discretisations Ω^h and Ω^{2h}. We will focus on 2-dimensional operators, but the extension to d-dimensions is simple. In §A.3, we discussed two possible grid discretisa-

tions for $\Omega \subset [0,1]^2$; vertex-centered and cell-centered. In this thesis, we only consider cell-centered discretisations as we are working with images of dimensions $n \times m$ and a cell-centered discretisation results in Ω^h of dimension $n \times m$, whereas a vertex-centered grid would be dimension $(n+1) \times (m+1)$. In Figure 2.22, we show a discretisation of Ω with $h = 1/8$ along with the coarser grids Ω^{2h}, Ω^{4h} and Ω^{8h}. We note that each pixel in Ω^{2h} contains 4 pixels on Ω^h. Therefore, if we want to transfer a quantity from the grid Ω^h to Ω^{2h} we must use a restriction operator which incorporates the information from the values on Ω^h to generate the value on Ω^{2h}. Similarly, to transfer a quantity from Ω^{2h} to Ω^h we must use an interpolation (or prolongation) operator to define the new values on Ω^h using the values on Ω^{2h}.



(a) Ω^h (b) Ω^{2h} (c) Ω^{4h} (d) Ω^{8h}

Figure 2.22: Various cell-centered grid discretisations for $h = 1/8$. Notice that each coarser grid has quadruple the cell size of the previous level.

There is a direct relation between the order of the transfer operator and the degree of the polynomial that is exactly interpolated by the corresponding interpolation rule.

Definition 2.3.2.1 (Order of restriction and interpolation [88])**.** If a restriction operator leaves all polynomials of degree $k-1$ invariant, then we say that the order of the operator is k, i.e. $r_{res} = k$. If an interpolation operator leaves all polynomials of degree $k-1$ invariant, then we say that it has order at least k, i.e. $r_{int} \geq k$.

Remark 2.3.2.2 (Boundary Values)**.** The schemes we give are all for interior grid points, those near the boundaries need to be adapted for the appropriate boundary conditions. These required modifications are important, but trivial, therefore are not

discussed here.

2.3.2.1 Restriction

To transfer a quantity from Ω^h to the coarser grid Ω^{2h} we can use a variety of restriction operators from the literature. We note that by using the cell-centered discretisation, each cell of the coarse grid Ω_{2h} contains within it 4 fine grid cells and each grid point of Ω_{2h} is surrounded by 4 grid points of Ω^h (see Figure 2.22). We will give the formulas for some popular restriction operators below. In all cases, the operator is given by I_h^{2h} and $\boldsymbol{x} = (x, y)$. Although we consider only 2D domains we note that d-dimensional restriction operators also exist, see [158].

Averaging Restriction [158]. The simplest, and most commonly used, restriction operator is the averaging restriction. This simply takes the values of the four surrounding grid points and averages them. In Figure 2.23 we display intuitively how this type of restriction works. This is a first-order operator [171].

$$\begin{aligned} u^{2h}(\boldsymbol{x}) &= I_h^{2h}(u^h(\boldsymbol{x}) \\ &= \frac{1}{4}\left[u^h\left(x-\frac{h}{2}, y-\frac{h}{2}\right) + u^h\left(x-\frac{h}{2}, y+\frac{h}{2}\right)\right. \\ &\quad \left. + u^h\left(x+\frac{h}{2}, y-\frac{h}{2}\right) + u^h\left(x+\frac{h}{2}, y+\frac{h}{2}\right)\right] \end{aligned} \tag{2.32}$$

The following operators are used when the PDE we are solving requires higher-order accuracy and are all second-order [173].

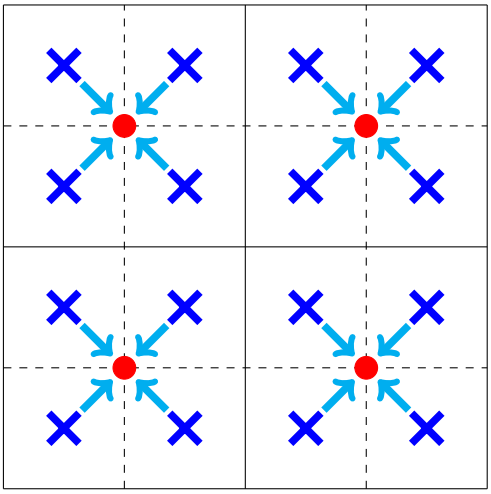


Figure 2.23: An intuitive view of averaging restriction. The grid points on Ω^h are shown by blue crosses and the grid points on Ω^{2h} are shown by red dots.

Bilinear/Full-weighting Restriction [158].

$$
\begin{aligned}
u^{2h}(\boldsymbol{x}) &= I_h^{2h}\left(u^h(\boldsymbol{x})\right) \\
&= \frac{9}{16}\left[u^h\left(x-\frac{h}{2},y-\frac{h}{2}\right)+u^h\left(x-\frac{h}{2},y+\frac{h}{2}\right)\right. \\
&\qquad \left.+u^h\left(x+\frac{h}{2},y-\frac{h}{2}\right)+u^h\left(x+\frac{h}{2},y+\frac{h}{2}\right)\right] \\
&+\frac{3}{16}\left[u^h\left(x-\frac{h}{2},y-3\frac{3h}{2}\right)+u^h\left(x+\frac{h}{2},y-3\frac{3h}{2}\right)\right. \\
&\qquad +u^h\left(x-3\frac{3h}{2},y-\frac{h}{2}\right)+u^h\left(x+3\frac{3h}{2},y-\frac{h}{2}\right) \\
&\qquad +u^h\left(x-3\frac{3h}{2},y+\frac{h}{2}\right)+u^h\left(x+3\frac{3h}{2},y+\frac{h}{2}\right) \\
&\qquad \left.+u^h\left(x-\frac{h}{2},y+3\frac{3h}{2}\right)+u^h\left(x+\frac{h}{2},y+3\frac{3h}{2}\right)\right] \\
&+\frac{1}{16}\left[u^h\left(x-\frac{3h}{2},y-\frac{3h}{2}\right)+u^h\left(x-\frac{3h}{2},y+\frac{3h}{2}\right)\right. \\
&\qquad \left.+u^h\left(x+\frac{3h}{2},y-\frac{3h}{2}\right)+u^h\left(x+\frac{3h}{2},y+\frac{3h}{2}\right)\right]
\end{aligned}
\tag{2.33}
$$

Wesseling/Khalil Restriction [173].

$$
\begin{aligned}
u^{2h}(\boldsymbol{x}) &= I_h^{2h}\left(u^h(\boldsymbol{x})\right) \\
&= \frac{1}{2}\left[u^h\left(x-\frac{h}{2},y+\frac{h}{2}\right)+u^h\left(x+\frac{h}{2},y-\frac{h}{2}\right)\right] \\
&+ \frac{3}{4}\left[u^h\left(x-\frac{h}{2},y-\frac{h}{2}\right)+u^h\left(x+\frac{h}{2},y+\frac{h}{2}\right)\right] \\
&+ \frac{1}{4}\left[u^h\left(x-\frac{3h}{2},y-\frac{3h}{2}\right)+u^h\left(x-\frac{3h}{2},y-\frac{h}{2}\right)\right. \\
&\quad +u^h\left(x-\frac{h}{2},y-\frac{3h}{2}\right)+u^h\left(x+\frac{h}{2},y+\frac{3h}{2}\right) \\
&\quad \left.+u^h\left(x+\frac{3h}{2},y+\frac{h}{2}\right)+u^h\left(x+\frac{3h}{2},y+\frac{3h}{2}\right)\right]
\end{aligned}
\tag{2.34}
$$

Kwak Restriction [101].

$$
\begin{aligned}
u^{2h}(\boldsymbol{x}) &= I_h^{2h}\left(u^h(\boldsymbol{x})\right) \\
&= \frac{1}{2}\left[u^h\left(x-\frac{h}{2},y-\frac{h}{2}\right)+u^h\left(x-\frac{h}{2},y+\frac{h}{2}\right)\right. \\
&\quad \left.+u^h\left(x+\frac{h}{2},y-\frac{h}{2}\right)+u^h\left(x+\frac{h}{2},y+\frac{h}{2}\right)\right] \\
&+ \frac{1}{4}\left[u^h\left(x-\frac{3h}{2},y-\frac{h}{2}\right)+u^h\left(x-\frac{3h}{2},y+\frac{h}{2}\right)\right. \\
&\quad +u^h\left(x-\frac{h}{2},y-\frac{3h}{2}\right)+u^h\left(x-\frac{h}{2},y+\frac{3h}{2}\right) \\
&\quad +u^h\left(x+\frac{h}{2},y-\frac{3h}{2}\right)+u^h\left(x+\frac{h}{2},y+\frac{3h}{2}\right) \\
&\quad \left.+u^h\left(x+\frac{3h}{2},y-\frac{h}{2}\right)+u^h\left(x+\frac{3h}{2},y+\frac{h}{2}\right)\right]
\end{aligned}
\tag{2.35}
$$

2.3.2.2 Interpolation

To transfer a quantity from Ω^{2h} to the finer grid Ω^h we can use a variety of interpolation operators from the literature. For brevity, we will only detail the bilinear interpolation operator, as this is what we exclusively use, but other operators can be found in [118].

We note that by using the cell-centered discretisation we create 4 grid points Ω^h for each single grid point on Ω^{2h}. Although we consider only 2D grids, we note that d-dimensional interpolation operators also exist, see [158].

Bilinear Interpolation [158]. The bilinear interpolation operator for cell-centered discretisation is given by:

$$u^h(\boldsymbol{x}) = I_{2h}^h \left(u^{2h}(\boldsymbol{x}) \right)$$

where

$$u^h(\boldsymbol{x}) = \begin{cases} \frac{1}{16}\left[9u^{2h}\left(x+\frac{h}{2},y-\frac{h}{2}\right)+3u^{2h}\left(x+\frac{h}{2},y+\frac{3h}{2}\right)\right. \\ \left.+3u^{2h}\left(x-\frac{3h}{2},y-\frac{h}{2}\right)+u^{2h}\left(x-\frac{3h}{2},y+\frac{3h}{2}\right)\right], & \text{for } \bullet, \\ \frac{1}{16}\left[9u^{2h}\left(x-\frac{h}{2},y-\frac{h}{2}\right)+3u^{2h}\left(x+\frac{3h}{2},y-\frac{h}{2}\right)\right. \\ \left.+3u^{2h}\left(x-\frac{h}{2},y+\frac{3h}{2}\right)+u^{2h}\left(x+\frac{3h}{2},y+\frac{3h}{2}\right)\right], & \text{for } \blacklozenge, \\ \frac{1}{16}\left[9u^{2h}\left(x-\frac{h}{2},y+\frac{h}{2}\right)+3u^{2h}\left(x-\frac{h}{2},y-\frac{3h}{2}\right)\right. \\ \left.+3u^{2h}\left(x+\frac{3h}{2},y+\frac{h}{2}\right)+u^{2h}\left(x+\frac{3h}{2},y-\frac{3h}{2}\right)\right], & \text{for } \blacksquare, \\ \frac{1}{16}\left[9u^{2h}\left(x+\frac{h}{2},y+\frac{h}{2}\right)+3u^{2h}\left(x+\frac{h}{2},y-\frac{3h}{2}\right)\right. \\ \left.+3u^{2h}\left(x-\frac{3h}{2},y+\frac{h}{2}\right)+u^{2h}\left(x-\frac{3h}{2},y-\frac{3h}{2}\right)\right], & \text{for } \pentagon. \end{cases} \tag{2.36}$$

In Figure 2.24, we display intuitively the interaction between the grid values on Ω^{2h} and Ω^h to generate the values on the finer grid.

Order of Interpolation and Restriction. Although we have some freedom to choose the restriction and interpolation operators, a result of Hemker [88] gives some restrictions. If we wish to solve a PDE of a given order accurately using multigrid then the author proves that the total order of restriction and interpolation operator must be of a certain order.

Theorem 2.3.2.3 (Required order of transfer operators [88])**.** Let r be the order of the operator $\mathcal{L}$ in the differential equation $\mathcal{L}u = f$. Let r_{res} and r_{int} be the orders of restriction and interpolation operators respectively. The order of the transfer

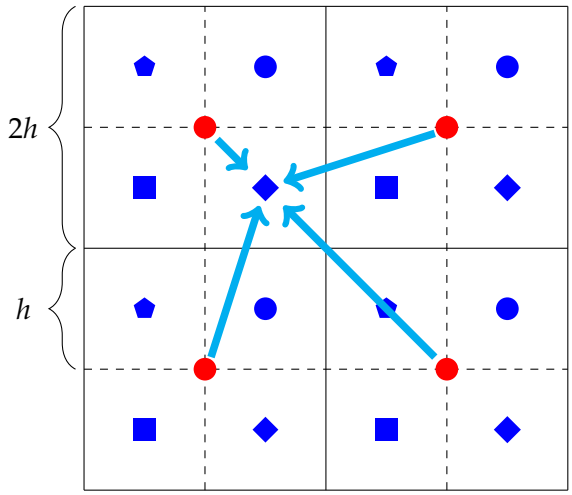


Figure 2.24: An intuitive view of how the node values on Ω^{2h} (the red dots) combine to form a value on Ω^h.

operators should fulfill

$$r_{res} + r_{int} > r.$$

Remark 2.3.2.4. In this thesis, we only solve PDEs of order 2, so we use bilinear interpolation ($r_{int} = 2$) and full-weighting restriction operators ($r_{res} = 2$) as they fulfill the above condition ($r_{res} + r_{int} = 4 > 2 = r$).

2.3.3. Linear Multigrid

In this section we will show how the components discussed so far (the smoother, restriction and interpolation operators and the coarse grid solver) can be combined to give the multigrid algorithm. We will first focus on linear multigrid, this being the algorithm we use for linear PDEs of the form $\mathcal{L}u = f$ for $\mathcal{L}$ a linear operator. The linearity of the operator gives us the "residual equation", which relates the residual at each iteration to the true algebraic error in the solution – this is not true in the non-linear case as we will discuss later.

2.3.3.1 Residual Equation

Recall that for the linear system

$$\mathcal{L}u = f.$$

the algebraic error of approximation $u^{(k)}$ is defined

$$e^{(k)} = u - u^{(k)}.$$

Definition 2.3.3.1 (Residual)**.** Let u be the exact solution of a linear system given by $\mathcal{L}u = f$. Suppose that we have an approximation $u^{(k)}$ which is the result of an iterative scheme after k steps. The residual is defined

$$r^{(k)} = f - \mathcal{L}u^{(k)}$$

The residual equation results from the following relationship

$$r^{(k)} = f - \mathcal{L}u^{(k)} = \mathcal{L}u - \mathcal{L}u^{(k)} = \mathcal{L}e^{(k)}.$$

This allows us to connect the residual of $u^{(k)}$ to the actual algebraic error – this is the holy grail of an algorithm, as once we know the algebraic error we can simply add it to our approximation to get the exact solution. Notice that we can calculate the residual exactly at each iteration as we know f, $\mathcal{L}$ and $u^{(k)}$ so this formula allows us to compute the algebraic error by

$$e^{(k)} = [\mathcal{L}]^{-1} r^{(k)}$$

and then setting $u = u^{(k)} + e^{(k)}$. However, notice that to obtain this we need to invert $\mathcal{L}$ and then solve for $e^{(k)}$ – this has the same computational difficulty as solving $\mathcal{L}u = f$ directly. This is where multigrid can be utilised. We aim to solve this equation for a coarser grid, with fewer grid points, and therefore it is not too computationally intensive. We discuss this idea in the next part.

2.3.3.2 Two-Grid Algorithm

In this section, we will introduce the two-grid algorithm, which involves only two levels of discretisation Ω^h and Ω^{2h}, this is the first building block to the multigrid algorithm. The two grid algorithm is shown in Algorithm 1.

Algorithm 1: Two-Grid Algorithm, $u^h \leftarrow \text{TGS}(u^h, \mathcal{L}^h, f^h, \nu_1, \nu_2, \text{Smoother})$

Pre-smoothing: Perform ν_1 iterations of the smoother: $u^h \leftarrow \text{Smoother}(u^h, f^h)$.

Coarse grid correction: Compute the residual: $r^h = f^h - \mathcal{L}^h u^h$.
Transfer the residual to Ω^{2h} by restriction: $r^{2h} = I_h^{2h} r^h$.
Compute: $e^{2h} = \left[\mathcal{L}^{2h}\right]^{-1} r^{2h}$.

Interpolation: Transfer the error to Ω^h by interpolation: $e^h = I_{2h}^h e^{2h}$.
Correct the fine grid approximation: $u^h = u^h + e^h$.

Post-smoothing: Perform ν_2 iterations of the smoother: $u^h \leftarrow \text{Smoother}(u^h, f^h)$.

The first step in this algorithm is the pre-smoothing, this ensures that the high-frequency components of the error are reduced and will not alias onto the coarser grid. The next step is the coarse grid correction, we compute the residual of the current approximation and transfer it to the coarse grid by restriction. We then solve the residual equation, discussed in the previous section. This gives the algebraic error discretised on the coarser grid. We then interpolate this up to the fine grid and update the solution. Finally, we perform some post-smoothing steps.

It is important to note some details of this algorithm. Firstly, we typically only use a small number of pre- and post-smoothing steps, $\nu_1, \nu_2 \leq 5$ is typical for a linear problem. Secondly, we notice that this algorithm requires computation of $\mathcal{L}^{2h}$. This can be obtained either simply by discretising the operator $\mathcal{L}$ to Ω^{2h} or using the Galerkin method $\mathcal{L}^{2h} = I_h^{2h} \mathcal{L}^h I_{2h}^h$ [158]. Finally, we must discuss how we actually solve the equation $e^{2h} = \left[\mathcal{L}^{2h}\right]^{-1} r^{2h}$. This can be accomplished by a direct solver such as Gaussian Elimination, Cholesky Decomposition, LU Factorisation, etc. or any of the methods given earlier in Table A.1. Alternatively, we may perform an iterative algorithm to convergence within some tolerance. The crucial part to notice is that although these algorithms may be computationally expensive, we now compute on Ω^{2h} rather than Ω^h so have fewer equations. If we suppose grid Ω^h has N grid points, then grid Ω^{2h} has $\frac{N}{4}$ grid points in 2D and $\frac{N}{8}$

grid points in 3D. Therefore, the computational savings of performing the majority of the work on the grid Ω^{2h} can be significant.

2.3.3.3 Multigrid Algorithm

The two-grid scheme can naturally be extended to allow the majority of the computations to be performed on $\Omega^{4h}, \Omega^{8h}, \ldots$. This is accomplished by the linear multigrid algorithm, shown in Algorithm 2.

Algorithm 2: Linear Multigrid Algorithm, $u^h \leftarrow$ LMG(u^h, $\mathcal{L}^h$, f^h, ν_1, ν_2, Smoother, level, max_level, η)

Pre-smoothing: Perform ν_1 iterations of the smoother: $u^h \leftarrow$Smoother(u^h, f^h).

Coarse grid correction: Compute the residual: $r^h = f^h - \mathcal{L}^h u^h$.

Transfer the residual to Ω^{2h} by restriction: $r^{2h} = I_h^{2h} r^h$.

if level = max_level **then**

Compute: $e^{2h} = \left[\mathcal{L}^{2h}\right]^{-1} r^{2h}$.

else

Do η cycles (steps) of $e^{2h} \leftarrow$ LMG($\mathbf{0}$, $\mathcal{L}^{2h}$, r^{2h}, ν_1, ν_2, Smoother, level+1, max_level, η).

end if

Interpolation: Transfer the error to Ω^h by interpolation: $e^h = I_{2h}^h e^{2h}$.

Correct the fine grid approximation: $u^h = u^h + e^h$.

Post-smoothing: Perform ν_2 iterations of the smoother: $u^h \leftarrow$Smoother(u^h, f^h).

This nested algorithm is very efficient and simple to implement. Note that we have also introduced a parameter η which permits non-standard cycling, i.e. if $\eta = 1$ this is a simple V-cycle where each grid is "visited" once, if $\eta = 2$ we have a W-cycle, etc, see [158].

Now that the linear multigrid algorithm has been considered, we will move onto the multigrid algorithm for non-linear problems. The important distinction being that the previous residual equation is no longer valid and we cannot find the algebraic error directly by inverting the operator.

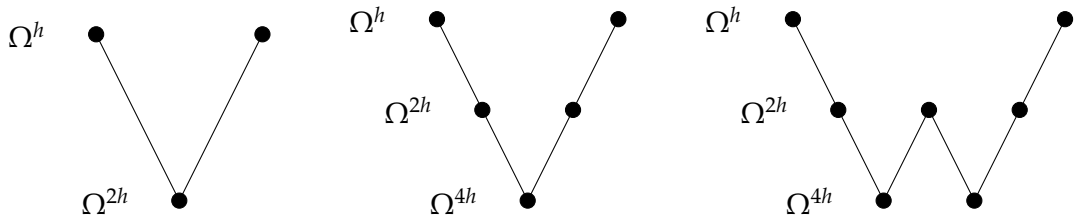


Figure 2.25: Diagram of (a) two-grid cycle, (b) three-grid V-cycle, (c) three grid W-cycle.

2.3.4. Non-Linear Multigrid

In this section, we will focus on the non-linear analogue for the linear multigrid algorithm. The key difference is we have a new residual equation which doesn't reduce as simply as in the linear case. We will focus on the non-linear multigrid algorithm called the Full Approximation Scheme (FAS). This algorithm is the focus of the work detailed in Chapter 5.

2.3.4.1 Non-Linear Residual Equation

For the non-linear system of equations given by $\mathcal{N}u = f$ we have the residual equation

$$r^{(k)} = f - \mathcal{N}u^{(k)} = \mathcal{N}u - \mathcal{N}u^{(k)} = \mathcal{N}\left(u^{(k)} + e^{(k)}\right) - \mathcal{N}u^{(k)},$$

where $r^{(k)}$ and $e^{(k)}$ are defined as for the linear case as the residual and algebraic error for $u^{(k)}$ respectively. This cannot be simplified nicely but we note that

$$u = [\mathcal{N}]^{-1}\left(r^{(k)} + \mathcal{N}u^{(k)}\right). \tag{2.37}$$

2.3.4.2 The Full Approximation Scheme

The Full Approximation Scheme (FAS) was introduced by Brandt [26]. It is a multigrid algorithm for non-linear problems and solves the equation given by (2.37) on a coarse grid. The FAS algorithm is shown in Algorithm 3.

We notice that the structure is largely similar to the linear multigrid algorithm, utilising

Algorithm 3: Full Approximation Scheme, $\phi^h \leftarrow$ FASMG(ϕ^h, $\mathcal{N}^h$, f^h, η, ν_1, ν_2, level, max_level, Smoother)

Pre-smoothing: Perform ν_1 iterations of the smoother: $\tilde{\phi}^h \leftarrow Smoother(\phi^h, f^h, \nu_1)$.

Coarse grid correction: Compute the residual: $r^h = f^h - N^h\tilde{\phi}^h$.
Transfer the residual to Ω^{2h} by restriction: $r^{2h} = I_h^{2h} r^h$.
Compute: $\phi^{2h} = I_h^{2h}\tilde{\phi}^h, \Phi^{2h} = \phi^{2h}, f^{2h} = N^{2h}\phi^{2h} + r^{2h}$.
if $level = max\_level$ **then**
  Compute the exact solution ϕ^{2h} of $N^{2h}\phi^{2h} = f^{2h}$
  on Ω^{2h} using an accurate solver.
else
  Perform η cycles (steps) of
  $\phi^{2h} \leftarrow$ FASMG(ϕ^{2h}, N^{2h}, f^{2h}, η, ν_1, ν_2, level+1, max_level, Smoother).
end if

Interpolation: Compute: $e^{2h} = \phi^{2h} - \Phi^{2h}$.
Transfer the error to Ω^h by interpolation: $e^h = I_{2h}^h e^{2h}$.
Correct the fine grid approximation: $\hat{\phi}^h = \tilde{\phi}^h + e^h$.

Post-smoothing: Perform ν_2 iterations of the smoother: $\phi^h \leftarrow Smoother(\hat{\phi}^h, f^h, \nu_2)$.

pre- and post-smoothing, coarse-grid correction and interpolation and restriction operators. There are several key differences to the linear multigrid algorithm. Firstly, we transfer our full solution between the grid levels rather than transferring just the error and residual. This is required as we are solving (2.37) on the coarsest grid then calculate the error as the difference between the result of this and the previous iterate of the solution (discretised to that grid). In this thesis, we focus only on the FAS multigrid algorithm but for completion, we remark that alternative algorithms do exist; namely Newton multigrid [90] and the non-linear multigrid method of Hackbusch [84]. Comparison of non-linear multigrid methods is made in [136].

To finish this section, we note that there is no simple multigrid "black box" solver for non-linear problems. Typically we need to tune and adjust the smoother that we use, the exact solver, the cycling structure and the transfer operators. For example, with a general non-linear problem, the smoothing rate in can be ≈ 1 where there are strong non-linearities in the PDE. In Chapter 5, we address this problem head-on by designing a non-standard smoother which guarantees a good smoothing rate for non-linear problems when compared to the standard smoothing schemes.

Chapter 3

A Convex Geodesic Selective Model for Image Segmentation

In this chapter, we introduce a new convex selective segmentation model which uses the geodesic distance from some user input as a penalty term in the formulation. It is shown in the results section that this model outperforms the current state-of-the-art. We will also give a proof that there exists a unique viscosity solution to the parabolic version of the PDE resulting from the minimisation of a whole class of segmentation functionals. In particular, we show that our proposed model is in this class. This chapter is based on the author's paper [137].

3.1. Introduction

If we consider a typical selective image segmentation model, they tend to have functionals of the following form

$$\mathcal{F} = \mathcal{F}_{\text{Regulariser}} + \mathcal{F}_{\text{Intensity Fitting}} + \mathcal{F}_{\text{Distance}}$$

where $\mathcal{F}_{\text{Regulariser}}$ is a regulariser such as TV or weighted-TV which ensures the solution has particular properties, e.g. has a short smooth boundary. $\mathcal{F}_{\text{Intensity Fitting}}$ ensures that the segmentation result has homogeneous intensity inside it and $\mathcal{F}_{\text{Distance}}$ encourages the

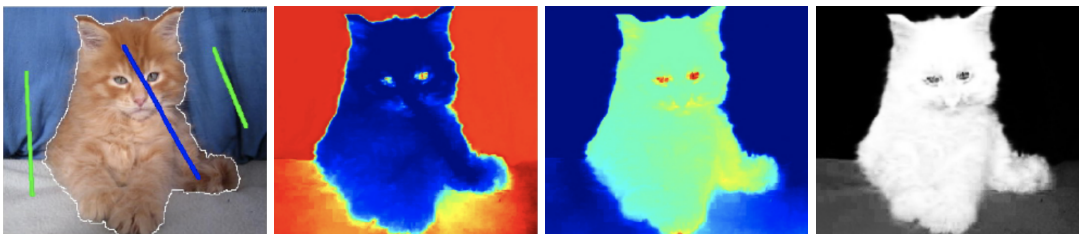
result to be near to some marker points which the user inputs. The models discussed earlier in §2.2.7 can all be formulated in this manner.

Inspired by this formulation, we propose using the edge-weighted geodesic distance from a marker set $\mathcal{M}$ (we call this simply the geodesic distance) as $\mathcal{F}_{\text{Distance}}$. We denote the set of k marker points by $\mathcal{M}$, i.e.

$$\mathcal{M} = \{\boldsymbol{x}_i \in \Omega, 1 \leq i \leq k\}$$

where $\Omega \subset \mathbb{R}^d$. This distance increases at edges in the image and is more intuitive for selective segmentation. The proposed model is given as a convex relaxed model with exact penalty term and we give a general existence and uniqueness proof for the viscosity solution to the PDE given by its Euler-Lagrange equation, which is also applicable to a whole class of PDEs arising in image segmentation.

We note that the use of geodesic distance for segmentation has been considered before [17, 133], but they use the geodesic distance function to "inform" a probabilistic function of whether a pixel belongs to the foreground or background of an image. See Figure 3.1.



(a) (b) (c) (d)

Figure 3.1: (a) The original image with scribbles for foreground and background, (b,c) The geodesic distance to foreground and background pixels respectively, (d) The calculated probability map, white pixels are in the foreground and black in the background. Figure taken from [133].

Here, we take a different approach, considering a variational approach by including geodesic distance as a standalone fitting term in the energy functional and using intensity fitting terms to ensure robustness.

In this chapter, we only consider 2D images, however, for completion we remark that 3D

segmentation models do exist [105, 176] and it is simple to extend the proposed model to 3D. The contributions of this chapter can be summarised as follows:

- We incorporate the geodesic distance as a distance penalty term within the variational framework.
- We propose a convex selective segmentation model using this penalty term and demonstrate how it can achieve results which cannot be achieved by other models.
- We improve the geodesic penalty term, focussing on improving robustness to noise and improving segmentation when object edges are blurred.
- We give an existence and uniqueness proof for the viscosity solution for the PDEs associated with a whole class of segmentation models (both global and selective).

We find that the proposed model gives accurate segmentation results for a wide range of parameters and, in particular, when segmenting the same objects from the same modality images, i.e. segmenting lungs from CT scans, the parameters are very similar from one image to the next to obtain accurate results. Therefore, this model may be used to assist in the preparation of large training sets for deep learning studies [129, 165, 166] that concern segmentation of particular objects from images.

The chapter is structured as follows; in §3.2, we discuss the geodesic distance penalty term, propose a new convex model and also address weaknesses in the naïve implementation of the geodesic distance term. In §3.3, we discuss the non-standard AOS scheme, introduced in [153], which we use to solve the model. In §3.4, we give an existence and uniqueness proof for a general class of PDEs arising in image segmentation, thereby showing that for a given initialisation the solution to our model is unique. In §3.5, we compare the results of the proposed model to other selective segmentation models, show that the proposed model is less parameter dependent than other models and is more robust to user input. Finally, in §3.6 we provide some concluding remarks.

3.2. Proposed Convex Geodesic Selective Model

In this section, we propose an improved selective model, based on the Spencer-Chen model (2.27), which uses the edge-weighted geodesic distance from the marker set $\mathcal{M}$ as the distance term, rather than the Euclidean distance. Increasing the distance when

edges in the image are encountered gives a more accurate reflection of the true similarity of pixels in an image from the marker set. We propose minimising the functional

$$\begin{aligned}\mathcal{F}_{CG}(u, c_1, c_2) = \mu \int_\Omega g(|\nabla z(x,y)|)|\nabla u| \, \mathrm{d}\Omega \\ + \int_\Omega \left[\lambda_1 (z(x,y) - c_1)^2 - \lambda_2 (z(x,y) - c_2)^2\right] u \, \mathrm{d}\Omega \\ + \theta \int_\Omega \mathcal{D}_M(x,y) u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu_\varepsilon(u) \, \mathrm{d}\Omega,\end{aligned} \tag{3.1}$$

where $\mathcal{D}_M(x,y)$ is the edge-weighted geodesic distance from the marker set.

Fixing the values c_1 and c_2 by (3.11 allows us to consider the minimisation of $\mathcal{F}_{CG}$ only over u which is, fortunately, a convex minimisation problem. This can be seen by reference to §2.2.6.3 as this is actually an extension of the convex relaxed Chan-Vese functional, with the addition of the term

$$\theta \int_\Omega \mathcal{D}_M(x,y) u \, \mathrm{d}\Omega.$$

This additional term is also convex in u, therefore the overall functional $\mathcal{F}_{CG}$ is also convex in u for fixed c_1 and c_2.

In Figure 3.2, we compare the normalised geodesic distance and the Euclidean distance from the same marker point (i.e. set $\mathcal{M}$ has one point in it); clearly the former gives a more intuitively correct distance penalty than the latter. We will refer to this proposed model as the Geodesic Model.

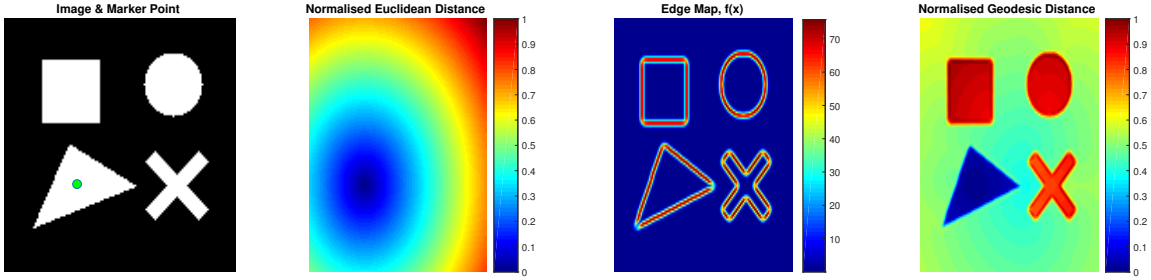


Figure 3.2: Comparison of distance measures. (i) Simple binary image with marker point; (ii) normalised Euclidean distance from marker point; (iii) edge map function $f(x)$ for the image; (iv) normalised geodesic distance from marker point.

3.2.1. Computing the Geodesic Distance Term $\mathcal{D}_M(x,y)$

The geodesic distance from the marker set $\mathcal{M}$ is given by $\mathcal{D}_M(x,y) = 0$ for $(x,y) \in \mathcal{M}$ and $\mathcal{D}_M(x,y) = \frac{\mathcal{D}^0_M(x,y)}{||\mathcal{D}^0_M(x,y)||_{L^\infty}}$ for $(x,y) \notin \mathcal{M}$, where $\mathcal{D}^0_M(x,y)$ is the solution of the following PDE

$$|\nabla \mathcal{D}^0_M(x,y)| = f(x,y), \qquad \mathcal{D}^0_M(x_0,y_0) = 0,\ (x_0,y_0) \in \mathcal{M}. \tag{3.2}$$

where $f(x,y)$ is defined later on with respect to the image contents.

If $f(x,y) \equiv 1$ (i.e. $|\nabla \mathcal{D}^0_M(x,y)| = 1$) then the distance penalty $\mathcal{D}_M(x,y)$ is simply the normalised Euclidean distance $\mathcal{D}_E(x,y)$ as used in the Spencer-Chen model (2.26). We have free rein to design $f(x,y)$ as we wish. Looking at the PDE in (3.2), we see that when $f(x,y)$ is small this results in a small gradient in our distance function and it is almost flat. When $f(x,y)$ is large, we have a large gradient in our distance map. In the case of selective image segmentation, we want small gradients in homogeneous areas of the image and large gradients at edges. If we set

$$f(x,y) = \varepsilon_{\mathcal{D}} + \beta_G |\nabla z(x,y)|^2 \tag{3.3}$$

this gives us the desired property that in areas where $|\nabla z(x,y)| \approx 0$, the distance function increases by some small $\varepsilon_{\mathcal{D}}$; here image $z(x,y)$ is scaled to $[0,1]$. At edges, $|\nabla z(x,y)|$ is large and the geodesic distance increases here. We set value of $\beta_G = 1000$ and $\varepsilon_{\mathcal{D}} = 10^{-3}$ throughout. In Figure 3.2, we see that the geodesic distance plot gives a low distance penalty on the triangle, which the marker indicates we would like segmented. There is a reasonable penalty on the background, and all other objects in the image have a very high distance penalty (as the geodesic to these points must cross two edges). This contrasts with the Euclidean distance, which gives a low distance penalty to some background pixels and maximum penalty to the pixels furthest away.

3.2.2. Comparing Euclidean and Geodesic Distance Terms

We briefly give some advantages of using the geodesic distance as a penalty term rather than Euclidean distance and a remark on the computational complexity for both distances.

1. **Parameter Robustness.** The Geodesic Model is more robust to the choice of the fitting parameter θ, as the penalty on the inside of the shape we want segmented is consistently small. It is only outside the shape where the penalty is large. Whereas with the Euclidean distance term we always have a penalty inside the shape we actually want to segment. This is due to the nature of the Euclidean distance which does not discriminate on intensity – this penalty can also be quite high if our marker set is small and doesn't cover the whole object.

2. **Robust to Marker Set Selection.** The geodesic distance term is far more robust to point selection, for example we can choose just one point inside the object we want to segment and this will give a nearly identical geodesic distance compared to choosing many more points. This is not true of the Euclidean distance term which is very sensitive to point selection and requires markers to be spread in all areas of the object you want to segment (especially at extrema of the object).

Remark 3.2.2.1 (Computational Complexity.)**.** The main concern of using the geodesic penalty term, which we obtain by solving PDE (3.2), would be that it takes a significant amount of time to compute compared to the Euclidean distance. However, using the fast marching algorithm of Sethian [148], the complexity of computing $\mathcal{D}_M(x,y)$ is $\mathcal{O}(N\log(N))$ for an image with N pixels. This is is only marginally more complex than computing the Euclidean distance which has $\mathcal{O}(N)$ complexity [117].

3.2.3. Improvements to Geodesic Distance Term

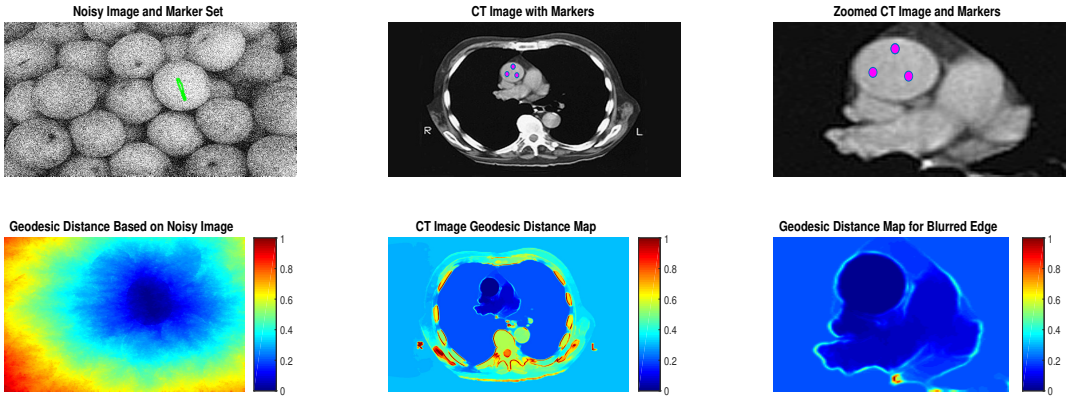
We now propose some modifications to the geodesic distance. Although the geodesic distance presents many advantages for selective image segmentation, we have three key disadvantages of this fitting term, which the Euclidean fitting term does not suffer.

1. **Not robust to noise.** The computation of the geodesic distance depends on $|\nabla z(x,y)|^2$ in $f(x,y)$ (see (3.2)). So, if an image contains a lot of noise, each noisy pixel appears as an edge and we get a misleading distance term. See Figure 3.3(a).

2. **Objects far from $\mathcal{M}$ with a low penalty.** As the geodesic distance only uses marker set $\mathcal{M}$ for its initial condition (see (3.2)), this can result in objects far from $\mathcal{M}$ having a low distance penalty, which is clearly not desired. See Figure 3.3(b).

3. **Blurred edges.** If we have two objects separated by a blurry edge and we have marker points only in one object, the geodesic distance will be low to the other object, as the edge penalty is weakly enforced for a blurry edge. We would desire low penalty inside the object with markers and a reasonable penalty in the joined object. See Figure 3.3(c).

We now propose solutions to each of these problems.



(a) Problem 1 (b) Problem 2 (c) Problem 3

Figure 3.3: Examples of the problems discussed in §3.2.3. In (a) we find that the geodesic distance shows a poor result if the image is noisy. In (b) we see that outside the patient has an unreasonably low distance penalty considering the Euclidean distance to these pixels. In (c) we show how the blurred edge under the aorta leads to the distance term being very low throughout the heart.

Problem 1: **Noise Robustness.** A naïve solution to the problem of noisy images would be to apply a Gaussian blur to $z(x,y)$ to remove the effect of the noise, so we change $f(x,y)$ to

$$\tilde{f}(x,y) = \varepsilon_{\mathcal{D}} + \beta_G |\nabla G_\sigma * z(x,y)|^2 \tag{3.4}$$

where G_σ is a Gaussian convolution with standard deviation σ. However, the effect of Gaussian convolution is that it also blurs edges in the image. This then gives us the same issues described in Problem 3. We see in Figure 3.4 column 3, that the Gaussian convolution reduces the sharpness of edges and this results in the geodesic distance being very similar in adjacent objects – therefore we see more pixels with high geodesic

distance. Our alternative to Gaussian blur is to consider anisotropic TV denoising. We

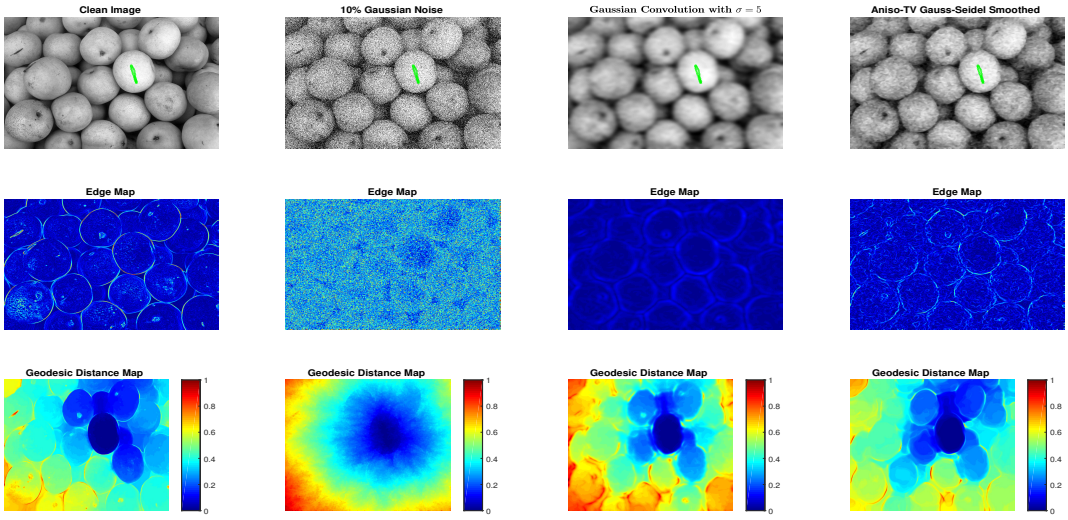


Figure 3.4: The edge maps and geodesic distance maps. (Left to right:) the clean image, the image with 10% Gaussian noise, the noisy image with Gaussian convolution applied ($\sigma = 5$) and for the noisy image with 100 iterations of anisotropic-TV Gauss-Seidel smoothing. The set $\mathcal{M}$ is shown on the top row in green, it is the same for each image.

refer the reader to [37, 131] for information on the model, here we just give the PDE which results from its minimisation:

$$\tilde{\mu}\nabla \cdot \left(g(|\nabla z(x,y)|) \frac{\nabla u}{|\nabla u|_{\varepsilon_2}} \right) + \iota(z(x,y) - u) = 0, \tag{3.5}$$

where $\tilde{\mu}, \iota$ are non-negative real parameters (we fix throughout $\tilde{\mu} = 10^{-3}, \iota = 5 \times 10^{-4}$), z is the given noisy image and where we replace $|\nabla u|$ with $|\nabla u|_{\varepsilon_2} = \sqrt{u_x^2 + u_y^2 + \varepsilon_2}$ to avoid zero denominator; we choose $\varepsilon_2 = 10^{-6}$ throughout. It is proposed to apply a relatively small number of cheap fixed point Gauss-Seidel iterations (between 100 and 200) to the discretised PDE. We cycle through all pixels (i, j) and update $u_{i,j}$ as follows

$$u_{i,j} = \frac{A_{i,j}u_{i+1,j} + B_{i,j}u_{i-1,j} + C_{i,j}u_{i,j+1} + D_{i,j}u_{i,j-1}}{A_{i,j} + B_{i,j} + C_{i,j} + D_{i,j} + \iota} \tag{3.6}$$

where

$$A_{i,j} = \frac{\tilde{\mu}}{h_x^2}\left(\frac{g(|\nabla z(x,y)|)}{|\nabla u_{\varepsilon_2}|}\right)_{i+1/2,j}, \quad B_{i,j} = \frac{\tilde{\mu}}{h_x^2}\left(\frac{g(|\nabla z(x,y)|)}{|\nabla u_{\varepsilon_2}|}\right)_{i-1/2,j},$$
$$C_{i,j} = \frac{\tilde{\mu}}{h_y^2}\left(\frac{g(|\nabla z(x,y)|)}{|\nabla u_{\varepsilon_2}|}\right)_{i,j+1/2}, \quad D_{i,j} = \frac{\tilde{\mu}}{h_y^2}\left(\frac{g(|\nabla z(x,y)|)}{|\nabla u_{\varepsilon_2}|}\right)_{i,j-1/2}. \tag{3.7}$$

We update all pixels once per iteration and solve the PDE in (3.2) with $f(x,y)$ replaced by

$$f_1(x,y) = \varepsilon_{\mathcal{D}} + \beta_G |\nabla S^k(z(x,y))|^2 \tag{3.8}$$

where S represents the Gauss-Seidel iterative scheme and k is the number of iterations performed (we choose $k = 100$ in our tests). In the final column of Figure 3.4 we see that the geodesic distance map more closely resembles that of the clean image than the Gaussian blurred map in column 3 and in Figure 3.5 we see that the segmentation results are qualitatively and quantitatively better using the anisotropic smoothing technique.

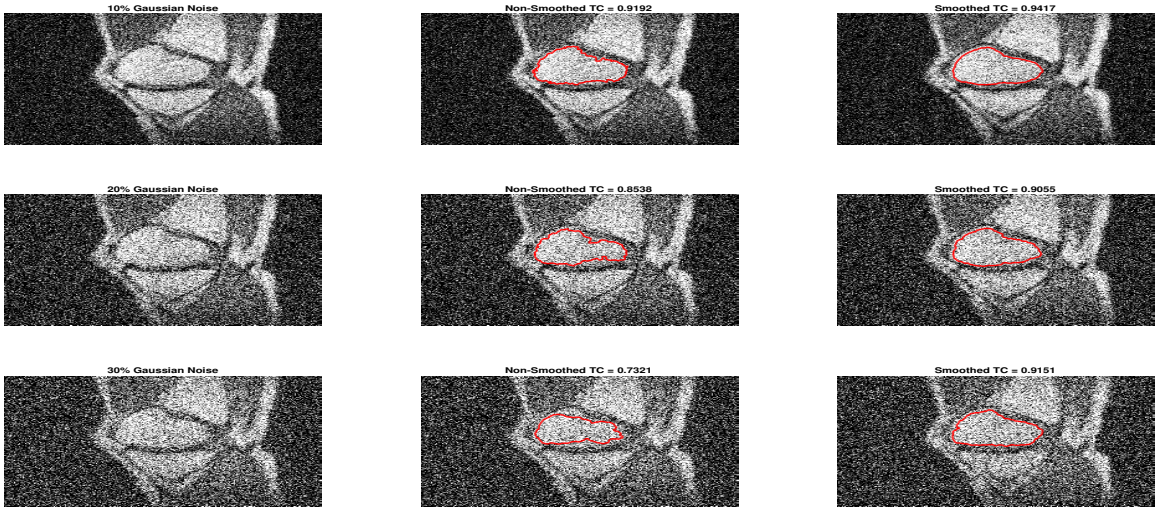


Figure 3.5: Segmentation results and Tanimoto Coefficients (see §3.5) for images with 10%, 20% and 30% Gaussian Noise with and without smoothing, $\lambda_1 = \lambda_2 = 5$, $\theta = 3$.

Problem 2: **Objects far from $\mathcal{M}$ with a low penalty.**

In Figure 3.3 column 2 we see that the geodesic distance to the outside of the patient

is lower than to their ribs. This is due to the fact that the region outside the body is homogeneous and there is almost zero distance penalty in this region. Similarly for Figure 3.4 column 4, the distance from the marker set to many surrounding objects is low, even though their Euclidean distance from the marker set is high. We wish to have the Euclidean distance $\mathcal{D}_E(x,y)$ incorporated somehow. Our solution is to modify the term $f_1(x,y)$ from (3.8) to

$$f_2(x,y) = \varepsilon_{\mathcal{D}} + \beta_G |\nabla S^k(z(x,y))|^2 + \vartheta \mathcal{D}_E(x,y). \tag{3.9}$$

In Figure 3.6 the effect of this is clear, as ϑ increases, the distance function resembles the Euclidean distance more. We use value $\vartheta = 10^{-1}$ in all experiments as it adds a reasonable penalty to pixels far from the marker set.

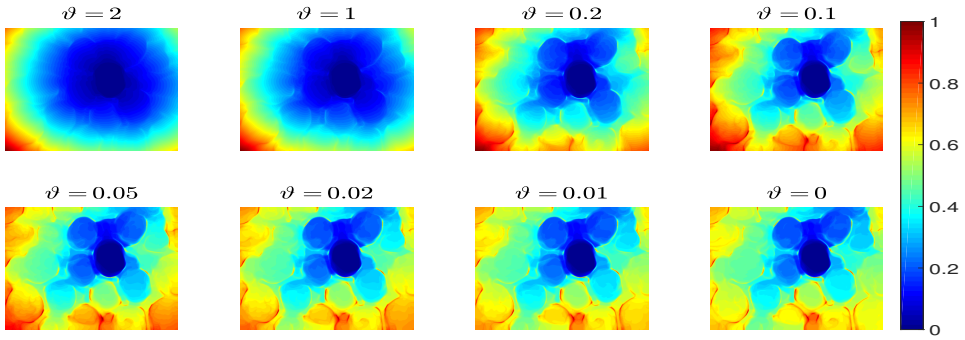


Figure 3.6: Displayed is $\mathcal{D}_M(x,y)$ using $f_2(x,y)$ for various ϑ values. The marker set is the same as that used in Figure 3.4.

Problem 3: **Blurred edges.**

If there are blurred edges between objects in an image, the geodesic distance will not increase significantly at this edge. Therefore the final segmentation result is liable to include unwanted objects. We look to address this problem through the use of anti-markers. These are markers which indicate objects that we do not want to segment, i.e. the opposite of marker points, we denote the set of anti-marker points by $\mathcal{AM}$. We propose to use a geodesic distance map from the set $\mathcal{AM}$ denoted by $\mathcal{D}_{AM}(x,y)$ which penalises pixels near to the set $\mathcal{AM}$ and doesn't add any penalty to those far away. We could naïvely choose $\mathcal{D}_{AM}(x,y) = 1 - \tilde{\mathcal{D}}_{GAM}(x,y)$ where $\tilde{\mathcal{D}}_{GAM}(x,y)$ is the normalised

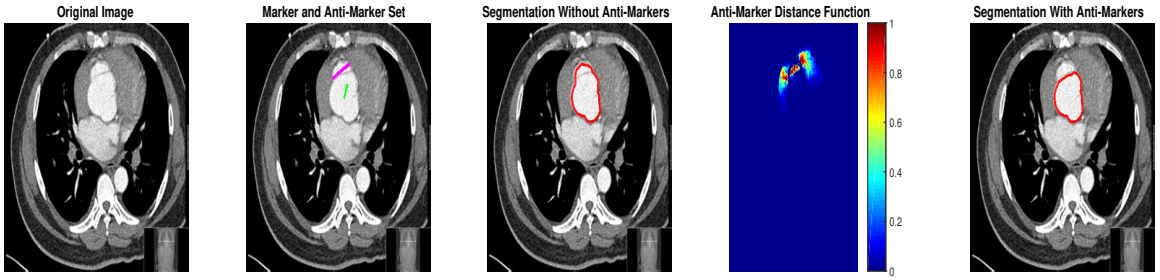


Figure 3.7: (Left to right:) original image, $\mathcal{M}$ (green) and $\mathcal{AM}$ (pink), segmentation result just using marker set, $\mathcal{D}_{AM}(x,y)$ using anti-markers, segmentation result using anti-markers. For these $\mu = 1, \lambda_1 = \lambda_2 = 5, \theta = 25$.

geodesic distance from $\mathcal{AM}$. However this puts a large penalty on those pixels inside the object we actually want to segment (as $\tilde{\mathcal{D}}_{GAM}(x,y)$ to those pixels is small). To avoid this problem, we propose the following anti-marker distance term

$$\mathcal{D}_{AM}(x,y) = \frac{\exp\left(-\tilde{\alpha}\tilde{\mathcal{D}}_{GAM}(x,y)\right) - \exp\left(-\tilde{\alpha}\right)}{1 - \exp\left(-\tilde{\alpha}\right)}$$

where $\tilde{\alpha}$ is a tuning parameter. We choose $\tilde{\alpha} = 200$ throughout. This distance term ensures rapid decay of the penalty away from the set $\mathcal{AM}$ but still enforces high penalty around the anti-marker set itself. See Figure 3.7 where a segmentation result with and without anti-markers is shown. As $\mathcal{D}_{AM}(x,y)$ decays rapidly from $\mathcal{AM}$, we do require that the anti-marker set be close to the blurred edge and away from the object we desire to segment.

3.2.4. The new model and its Euler-Lagrange equation

The Proposed Geodesic Model. Putting the above 3 ingredients together, we propose the model

$$\min_{u,c_1,c_2}\Big\{\mathcal{F}_{GEO}(u,c_1,c_2) = \int_\Omega \left[\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2\right] u \,\mathrm{d}\Omega + \mu\int_\Omega g(|\nabla z(x,y)|)|\nabla u|\,\mathrm{d}\Omega + \theta\int_\Omega \mathcal{D}_G(x,y)u\,\mathrm{d}\Omega + \alpha\int_\Omega \nu_\varepsilon(u)\,\mathrm{d}\Omega\Big\}, \tag{3.10}$$

where $\mathcal{D}_G(x,y) = \left(\mathcal{D}_M(x,y) + \mathcal{D}_{AM}(x,y)\right)/2$ and $\mathcal{D}_M(x,y)$ is the geodesic distance from

the marker set $\mathcal{M}$ and z is the given image. We compute $\mathcal{D}_M(x,y)$ using (3.2) where $f(x,y) = f_2(x,y)$ defined in (3.9). Using Calculus of Variations, solving (3.10) with respect to $c_1,\ c_2$, with u fixed, leads to

$$c_1(u) = \frac{\int_\Omega u \cdot z(x,y)\,\mathrm{d}\Omega}{\int_\Omega u\,\mathrm{d}\Omega}, \qquad c_2(u) = \frac{\int_\Omega (1-u)\cdot z(x,y)\,\mathrm{d}\Omega}{\int_\Omega (1-u)\,\mathrm{d}\Omega}, \tag{3.11}$$

and the minimisation with respect to u (with c_1 and c_2 fixed) gives the PDE

$$\begin{aligned}\mu\nabla\cdot\left(g(|\nabla z(x,y)|)\frac{\nabla u}{|\nabla u|_{\varepsilon_2}}\right) - \left[\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2\right] \\ - \theta\mathcal{D}_G(x,y) - \alpha v_\varepsilon'(u) = 0\end{aligned} \tag{3.12}$$

in Ω, where we replace $|\nabla u|$ with $|\nabla u|_{\varepsilon_2} = \sqrt{u_x^2 + u_y^2 + \varepsilon_2}$ to avoid zero denominator; we choose $\varepsilon_2 = 10^{-6}$ throughout. We also have Neumann boundary conditions $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ on $\partial\Omega$ where $\boldsymbol{n}$ is the outward unit normal vector.

Next we discuss a numerical scheme for solving this PDE (3.12). However it should be remarked that updating $c_1(u), c_2(u)$ should be done as soon as u is updated; practically c_1, c_2 converge very quickly since the object intensity c_1 does not change much.

3.3. An Additive Operator Splitting Algorithm

Additive Operator Splitting (AOS) is a widely used method [75, 113, 170] as seen from more recent works [10, 11, 12, 13, 135, 153] on the diffusion type equation such as

$$\frac{\partial u}{\partial t} = \mu\nabla\cdot(G(u)\nabla u) - f. \tag{3.13}$$

AOS allows us to split the two-dimensional problem into two one-dimensional problems, which we solve and then combine. Each one-dimensional problem gives rise to a tridiagonal system of equations which can be solved efficiently, hence AOS is a very efficient method for solving diffusion-like equations. AOS is a semi-implicit method and permits far larger time-steps than the corresponding explicit schemes would. Hence AOS is more

stable than an explicit method [170]. We rewrite the above equation as

$$\frac{\partial u}{\partial t} = \mu\Big(\partial_x\left(G(u)\partial_x u\right) + \partial_y\left(G(u)\partial_y u\right)\Big) - f.$$

and after discretisation, we can reformulate this as the AOS scheme [170]

$$u^{k+1} = \frac{1}{2}\sum_{\ell=1}^{2}\Big(I - 2\tau\mu A_\ell(u^k)\Big)^{-1}\Big(u^k + \tau f\Big)$$

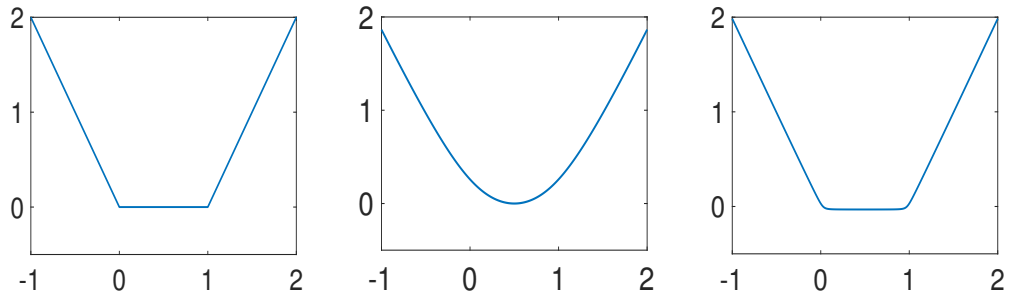
where τ is the time-step, $A_1(u) = \partial_x(G(u)\partial_x)$ and $A_2(u) = \partial_y(G(u)\partial_y)$. For notational convenience we write $G = G(u)$. The matrix $A_1(u)$ can be obtained as follows

$$\Big(A_1(u^k)u^{k+1}\Big)_{i,j} = \Big(\partial_x\Big(G\partial_x u^{k+1}\Big)\Big)_{i,j} = \left(\frac{G_{i+\frac{1}{2},j}}{h_x^2}\right)u_{i+1,j}^{k+1} + \left(\frac{G_{i-\frac{1}{2},j}}{h_x^2}\right)u_{i-1,j}^{k+1} - \left(\frac{G_{i+\frac{1}{2},j} + G_{i-\frac{1}{2},j}}{h_x^2}\right)u_{i,j}^{k+1}$$

and similarly to [135, 153], for the half points in G we take the average of the surrounding pixels, e.g. $G_{i+\frac{1}{2},j} = \frac{G_{i+1,j}+G_{i,j}}{2}$. Therefore we must solve two tridiagonal systems to obtain u^{k+1}, the Thomas algorithm allows us to solve each of these efficiently [170]. The AOS method described here assumes f does not depend on u, however in our case it depends on $\nu_\varepsilon'(u)$ (see (3.12)) which has jumps around 0 and 1, so the algorithm has stability issues. This was noted in [153] and the authors adapted the formulation of (3.12) to offset the changes in f. Here we repeat their arguments for adapting AOS when the exact penalty term $\nu_\varepsilon'(u)$ is present (we refer to Figures 3.8 and 3.9 for plots of the penalty function and its derivative, respectively).

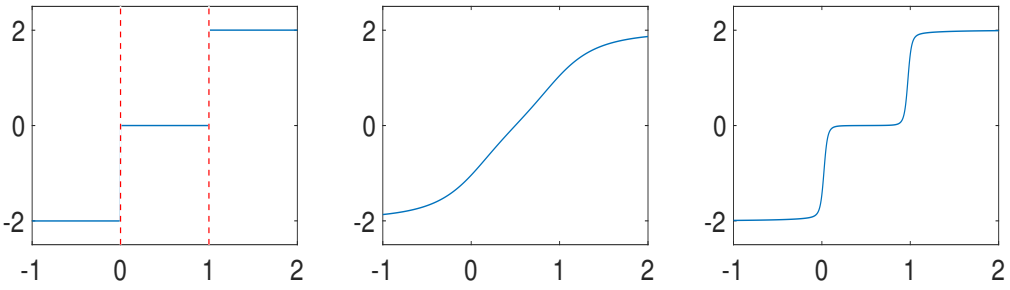
The main consideration is to extract a linear part out of the non-linearity in $f = f(u)$. If we evaluate the Taylor expansion of $\nu_\varepsilon'(u)$ around $u = 0$ and $u = 1$ and group the terms into the constant and linear components in u we find that the coefficient of the linear term is the same in both expansions. We denote this linear coefficient as b from now on. Therefore, for a change in u of δu around $u = 0$ and $u = 1$, we can approximate the change in $\nu_\varepsilon'(u)$ by $b \cdot \delta u$. To focus on the jumps, define the interval in which $\nu_\varepsilon'(u)$ jumps as

$$I_\zeta := [0 - \zeta, 0 + \zeta] \cup [1 - \zeta, 1 + \zeta]$$



(a) $\nu(u)$. (b) $\nu_\varepsilon(u)$ for $\varepsilon = 1$. (c) $\nu_\varepsilon(u)$ for $\varepsilon = 0.1$.

Figure 3.8: (a) The exact penalty function $\nu(u)$ and (b,c) $\nu_\varepsilon(u)$ for different ε values.



(a) $\nu'(u)$. (b) $\nu'_\varepsilon(u)$ for $\varepsilon = 1$. (c) $\nu'_\varepsilon(u)$ for $\varepsilon = 0.1$.

Figure 3.9: (a) $\nu'(u)$ (discontinuities shown in red) and (b,c) $\nu'_\varepsilon(u)$ for different ε values.

and refine the linear function by

$$\tilde{b}^k_{i,j} = \begin{cases} b, & u^k_{i,j} \in I_\zeta \\ 0, & else. \end{cases}$$

Using these we can now offset the change in $\nu'_\varepsilon(u^k)$ by changing the formulation (3.13) to

$$\frac{\partial u}{\partial t} = \mu \nabla \cdot (G(u)\nabla u) - \alpha \tilde{b}^k u + \left[\alpha \tilde{b}^k u - f\right]$$

and discretising the left-hand side we get

$$u^{k+1} = u^k + \tau\mu \nabla \cdot (G(u^k)\nabla u^{k+1}) - \tau\alpha \tilde{b}^k u^{k+1} + \tau\left[\alpha \tilde{b}^k u^k - f^k\right]$$

which, following the derivation in [153], can be reformulated as

$$u^{k+1} = \frac{1}{2}\sum_{\ell=1}^{2} \underbrace{\left(I + \tilde{B}^k - 2\tau\mu A_\ell\left(u^k\right)\right)}_{Q_1}^{-1} \left(\left(I + \tilde{B}^k\right)u^k + \tau f^k\right)$$

where $\tilde{B}^k = \text{diag}(\tau\alpha\tilde{b}^k)$. We note that Q_1 is invertible as it is strictly diagonally dominant. This scheme improves on (3.13) as now, changes in f^k are damped. However, it is found in [153] that although this scheme does satisfy most of the discrete scale-space conditions of Weickert [170] (which guarantee convergence of the scheme), it does not satisfy all of them. In particular the matrix Q_1 doesn't have unit row sum and is not symmetrical.

Remark 3.3.0.1. In Weickert [170] the authors require the symmetry and unit row sum for the inverse, i.e. Q_1^{-1} and Q_2^{-1}. However proving these hold for non-singular Q_1 and Q_2 is equivalent.

- The inverse of a symmetric matrix is symmetric, as $AA^{-1} = Id = Id^T(AA^{-1})^T \Rightarrow AA^{-1} = (A^{-1})^T A^T \Rightarrow Id = AA^{-1} = (A^{-1})^T A \Rightarrow A^{-1} = (A^{-1})^T$.
- The inverse of a matrix with unit row sum also has unit row sum, as if we suppose $w = [1, \ldots, 1]^T$ then $Aw = w \Rightarrow A^{-1}w = w$. So if A has unit row sum, then so does A^{-1}.

The authors adapt the scheme above to the equivalent

$$u^{k+1} = \frac{1}{2}\sum_{\ell=1}^{2} \underbrace{\left(I - 2\tau\mu\left(I + \tilde{B}^k\right)^{-1} A_\ell\left(u^k\right)\right)}_{Q_2}^{-1} \left(u^k + \tau\left(I + \tilde{B}^k\right)^{-1} f^k\right), \tag{3.14}$$

where the matrix Q_2 does have unit row sum, however the matrix is not always symmetrical. We can guarantee convergence for $\zeta = 0.5$ (in which case Q_2 must be symmetrical) but we desire to use a small ζ to give a small interval I_ζ.

We find experimentally that convergence is achieved for any small value of ζ, this is due to the fact that at convergence the solution u is almost binary [44]. Therefore, although initially Q_2 is asymmetrical at some pixels, at convergence all pixels have values which fall within I_ζ and $I + \tilde{B}^k$ is a matrix with all diagonal entries $1 + \tau\alpha b$. Therefore we

find that at convergence Q_2 is symmetrical and the discrete scale-space conditions are all satisfied. In all of our tests we fix $\zeta = 0.01$.

Smoother 1: Solution of the Geodesic Model

Set μ, λ, θ. Compute $g(|\nabla z(x,y)|) = \frac{1}{1+\beta_G|\nabla z(x,y)|^2}$ and $\mathcal{D}_G(x,y) = \frac{\mathcal{D}_G^0(x,y)}{||\mathcal{D}_G^0(x,y)||_{L^\infty}}$, with $\mathcal{D}_G^0(x,y)$ the solution of (3.2). Initialise $u^{(0)}$ arbitrarily.
for *iter* $= 1$ to *max_iterations* **do**
  Calculate c_1 and c_2 using (3.11).
  Calculate $r = \lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2 + \theta\mathcal{D}_G$.
  Set $\alpha = ||r||_{L^\infty}$.
  Calculate $f^k = r + \alpha v_\varepsilon'(u^k)$.
  Update u^k to u^{k+1} using the AOS scheme (3.14).
end for
$u^* \leftarrow u^k$.

3.4. Existence and Uniqueness of the Viscosity Solution

In this section we use the viscosity solution framework and the work of Ishii and Sato [94] to prove that, for a class of PDEs in image segmentation, the solution exists and is unique for a given initialisation. In particular, we prove the existence and uniqueness of the viscosity solution for the PDE which is determined by the Euler-Lagrange equation for the Geodesic Model. Throughout, we will assume Ω is a bounded domain with C^1 boundary.

Definition 3.4.0.1 (Viscosity Solutions)**.** We give several definitions taken from [56] which build to defining a viscosity solution.

- A PDE $F(x,u,Du,D^2u) = 0$ in a domain Ω is defined to be **degenerate elliptic** if for any two symmetric matrices X and Y such that $Y - X$ is positive definite, and any values of $x \in \Omega$, $u \in \mathbb{R}$ and $p \in \mathbb{R}^n$, we have the inequality $F(x,u,p,X) \geq F(x,u,p,Y)$.
- An upper semicontinuous function u in Ω is defined to be a **subsolution** of a degenerate elliptic equation in the viscosity sense if for any point $x_0 \in \Omega$ and

any C^2 function ϕ such that $\phi(x_0) = u(x_0)$ and $\phi \geq u$ in a neighborhood of x_0, we have $F(x_0, \phi(x_0), D\phi(x_0), D^2\phi(x_0)) \leq 0$.

- A lower semicontinuous function u in Ω is defined to be a **supersolution** of a degenerate elliptic equation in the viscosity sense if for any point $x_0 \in \Omega$ and any C^2 function ϕ such that $\phi(x_0) = u(x_0)$ and $\phi \leq u$ in a neighborhood of x_0, we have $F(x_0, \phi(x_0), D\phi(x_0), D^2\phi(x_0)) \geq 0$.
- A continuous function u is a **viscosity solution** of the PDE if it is both a supersolution and a subsolution.

From the work of [56, 94], we have the following Theorem for analysing the solution of a partial differential equation of the form $F(\boldsymbol{x}, u, Du, D^2u) = 0$ where $F : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathscr{M}^n \to \mathbb{R}$, $\mathscr{M}^n$ is the set of $n \times n$ symmetric matrices, Du is the gradient of u and D^2u is the Hessian of u. For simplicity, and in a slight abuse of notation, we use $x := \boldsymbol{x}$ for the vector of a general point in $\mathbb{R}^n$.

Theorem 3.4.0.2 (Theorem 3.1 [94])**.** Assume that the following conditions (C1)–(C2) and (I1)–(I7) hold. Then for each $u_0 \in C(\overline{\Omega})$ there is a unique viscosity solution $u \in C([0,T) \times \overline{\Omega})$ of (3.15) and (3.16) satisfying (3.17).

$$\frac{\partial u}{\partial t} + F(t, x, u, Du, D^2u) = 0 \quad \text{in} \quad Q = (0,T) \times \Omega, \tag{3.15}$$

$$B(x, Du) = 0 \quad \text{in} \quad S = (0,T) \times \partial\Omega, \tag{3.16}$$

$$u(0, x) = u_0(x) \quad \text{for} \quad x \in \overline{\Omega}. \tag{3.17}$$

Conditions (C1)–(C2).

(C1) $F(t, x, u, p, X) \leq F(t, x, v, p, X)$ for $u \leq v$.

(C2) $F(t, x, u, p, X) \leq F(t, x, u, p, Y)$ for $X, Y \in \mathscr{M}^n$ and $Y \leq X$.

Conditions (I1)–(I7). Assume Ω is a bounded domain in $\mathbb{R}^n$ with C^1 boundary.

(I1) $F \in C\left([0,T] \times \overline{\Omega} \times \mathbb{R} \times (\mathbb{R}^n \backslash \{0\}) \times \mathscr{M}^n\right)$.

(I2) There exists a constant $\gamma \in \mathbb{R}$ such that for each

$$(t, x, p, X) \in [0, T] \times \overline{\Omega} \times (\mathbb{R}^n \backslash \{0\}) \times \mathscr{M}^n$$

the function $u \mapsto F(t, x, u, p, X) - \gamma u$ is non-decreasing on $\mathbb{R}$.

(I3) F is continuous at $(t, x, u, 0, 0)$ for any $(t, x, u) \in [0, T] \times \overline{\Omega} \times \mathbb{R}$ in the sense that

$$-\infty < F_*(t, x, u, 0, 0) = F^*(t, x, u, 0, 0) < \infty$$

holds. Here F^* and F_* denote, respectively, the upper and lower semi-continuous envelopes of F, which are defined on $[0, T] \times \overline{\Omega} \times \mathbb{R} \times \mathbb{R}^n \times \mathscr{M}^n$.

(I4) $B \in C\left(\mathbb{R}^n \times \mathbb{R}^n\right) \cap C^{1,1}\left(\mathbb{R}^n \times (\mathbb{R}^n \backslash \{0\})\right)$, where $C^{1,1}$ is the Hölder functional space.

(I5) For each $x \in \mathbb{R}^n$ the function $p \mapsto B(x, p)$ is positively homogeneous of degree one in p, i.e. $B(x, \lambda p) = \lambda B(x, p)$ for all $\lambda \geq 0$ and $p \in \mathbb{R}^n \backslash \{0\}$.

(I6) There exists a positive constant Θ such that $\langle \boldsymbol{n}(x), D_p B(x, p) \rangle \geq \Theta$ for all $x \in \partial\Omega$ and $p \in \mathbb{R}^n \backslash \{0\}$. Here $\boldsymbol{n}(x)$ denotes the unit outward normal vector of Ω at $x \in \partial\Omega$.

(I7) For each $R > 0$ there exists a non-decreasing continuous function $\omega_R : [0, \infty) \to [0, \infty)$ satisfying $\omega_R(0) = 0$ such that if $X, Y \in \mathscr{M}^n$ and $\mu_1, \mu_2 \in [0, \infty)$ satisfy

$$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \leq \mu_1 \begin{bmatrix} I & -I \\ -I & I \end{bmatrix} + \mu_2 \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \tag{3.18}$$

then

$$\begin{aligned} F(t, x, u, p, X) - F(t, y, u, q, -Y) \geq -\omega_R\Big(&\mu_1 \left(|x - y|^2 + \rho(p, q)^2\right) + \mu_2 + |p - q| \\ &+ |x - y| \left(\max(|p|, |q|) + 1\right)\Big) \end{aligned}$$

for all $t \in [0, T], x, y \in \overline{\Omega}, u \in \mathbb{R}$, with $|u| \leq R$, $p, q \in \mathbb{R}^n \backslash \{0\}$ and $\rho(p, q) = \min\left(\frac{|p-q|}{\min(|p|, |q|)}, 1\right)$.

3.4.1. Existence and Uniqueness for the Geodesic Model

We now prove that, for a given initialisation of u, there exists a unique solution for the PDE (3.12) resulting from the minimisation of the functional for the Geodesic Model (3.10).

Remark 3.4.1.1. It is important to note that although the values of c_1 and c_2 depend on u, they are fixed when we solve the PDE for u and therefore the problem is a local one and Theorem 3.4.0.2 can be applied. Once we update c_1 and c_2, using the updated u, then we fix them again and apply Theorem 3.4.0.2. In practice, as we near convergence, we find c_1 and c_2 stabilise so we typically stop updating c_1 and c_2 once the change in both values is below a tolerance.

To apply the above theorem to the proposed model (3.12), the key step will be to verify the nine conditions. First, we multiply (3.12) by the factor $|\nabla u|_{\varepsilon_2}$, obtaining the non-linear PDE

$$\begin{aligned} &-\mu|\nabla u|_{\varepsilon_2}\nabla\cdot\left(G(x,\nabla z)\frac{\nabla u}{|\nabla u|_{\varepsilon_2}}\right) \\ &+|\nabla u|_{\varepsilon_2}\Big[\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2+\theta\mathcal{D}_G(x,y)+\alpha\nu_\varepsilon'(u)\Big]=0 \end{aligned} \tag{3.19}$$

where $G(x,\nabla z)=g(|\nabla z(x,y)|)$.

Remark 3.4.1.2. We multiply (3.12) by the factor $|\nabla u|_{\varepsilon_2}$ as it permits us to express the PDE in the simplified form of (3.20) and (3.21) below. This does not alter the solution of the PDE as $|\nabla u|_{\varepsilon_2}\neq 0$.

We can rewrite this as

$$F(x,u,p,X)=-\mu\,\mathrm{trace}\,(A(x,p)X)-\mu\langle\nabla G(x,\nabla z),p\rangle+|p|k(u)+|p|f(x)=0 \tag{3.20}$$

where $f(x)=\lambda_1(z(x)-c_1)^2-\lambda_2(z(x)-c_2)^2+\theta\mathcal{D}_G$, $k(u)=\alpha\nu_\varepsilon'(u)$, $p=(p_1,p_2)=|\nabla u|_{\varepsilon_2}$, X is the Hessian of u and

$$A(x,p)=\begin{bmatrix} G(x,\nabla z)\frac{p_2^2}{|p|^2} & -G(x,\nabla z)\frac{p_1p_2}{|p|^2} \\ -G(x,\nabla z)\frac{p_1p_2}{|p|^2} & G(x,\nabla z)\frac{p_1^2}{|p|^2}\end{bmatrix} \tag{3.21}$$

Theorem 3.4.1.3 (Theory for the Geodesic Model)**.** The parabolic PDE

$$\frac{\partial u}{\partial t} + F(t, x, u, Du, D^2u) = 0$$

with $u_0 = u(0, x) \in C(\overline{\Omega})$, F as defined in (3.20) and Neumann boundary conditions has a unique solution $u = u(t, x)$ in $C([0, T) \times \overline{\Omega})$.

Proof. By Theorem 3.4.0.2, it remains to verify that F satisfies (C1)–(C2) and (I1)–(I7). We will show that each of the conditions is satisfied. Most are simple to show, the exception being (I7) which is non-trivial.

(C1): Equation (3.20) only has dependence on u in the term $k(u)$, we therefore have a restriction on the choice of k, requiring $k(v) \geq k(u)$ for $v \geq u$. This is satisfied for $k(u) = \alpha \nu'_\varepsilon(u)$ with $\nu'_\varepsilon(u)$ defined as in (2.18).

(C2): We find for arbitrary $s = (s_1, s_2) \in \mathbb{R}^2$ that $s^T A(x, p)s \geq 0$ and so $A(x, p) \geq 0$. It follows that $-\text{trace}(A(x, p)X) \leq -\text{trace}(A(x, p)Y)$, therefore this condition is satisfied.

(I1): $A(x, p)$ is only singular at $p = 0$, however it is continuous elsewhere and satisfies this condition.

(I2): In F the only term which depends on u is $k(u) = \alpha \nu'_\varepsilon(u)$. With $\nu'_\varepsilon(u)$ defined as in (2.18), in the limit $\varepsilon \to 0$ this function is a step function from -2 on $(\infty, 0)$, 0 on $[0, 1]$ and 2 on $(0, \infty)$. So we can choose any constant $\varepsilon < -2$. With $\varepsilon \neq 0$ there is smoothing at the end of the intervals, however there is still a lower bound on L for $\nu'_\varepsilon(u)$ and we can choose any constant $\gamma < L$.

(I3): F is continuous at $(x, 0, 0)$ for any $x \in \Omega$ because $F^*(x, 0, 0) = F_*(x, 0, 0) = 0$. Hence this condition is satisfied.

(I4): The Euler-Lagrange equations give Neumann boundary conditions

$$B(x, \nabla u) = \frac{\partial u}{\partial \boldsymbol{n}} = \boldsymbol{n} \cdot \nabla\, u = \langle \boldsymbol{n}, \nabla u \rangle = 0$$

on $\partial\Omega$, where $\boldsymbol{n}$ is the outward unit normal vector, and we see that

$$B(x, \nabla u) \in C^{1,1}\left(\mathbb{R}^n \times \mathbb{R}^n \backslash \{0\}\right)$$

and therefore this condition is satisfied.

(I5): By the definition above, $B(x, \lambda\nabla u) = \langle \boldsymbol{n}, \lambda\nabla u\rangle = \lambda\langle \boldsymbol{n}, \nabla u\rangle = \lambda B(x, \nabla u)$. So this condition is satisfied.

(I6): As before we can use the definition, $\langle \boldsymbol{n}(x), D_p B(x,p)\rangle = \langle \boldsymbol{n}(x), \boldsymbol{n}(x)\rangle = |\boldsymbol{n}(x)|^2$. So we can choose $\Theta = 1$ and the condition is satisfied.

(I7): This is the most involved condition to prove and uses many other results. For clarity of the overall chapter, we postpone the proof to Appendix A. □

3.4.2. Generalisation to Other Related Models

Theorems 3.4.0.2 and 3.4.1.3 can be generalised to a few other models. This amounts to writing each model as a PDE of the form (3.20) where $k(u)$ is monotone and $f(x), k(u)$ are bounded. This is summarised in the following Corollary:

Corollary 3.4.2.1. Assume that c_1 and c_2 are fixed, with the terms $f(x)$ and $k(u)$ respectively defined as follows for a few related models:

- **Chan-Vese [46]**: $f(x) = f_{CV}(x) := \lambda_1(z(x) - c_1)^2 - \lambda_2(z(x) - c_2)^2$, $k(u) = 0$.
- **Chan-Vese (Convex) [44]**: $f(x) = f_{CV}(x)$, $k(u) = \alpha v'_\varepsilon(u)$.
- **Geodesic Active Contours [36] and Gout et al. [105]**: $f(x) = 0$, $k(u) = 0$.
- **Nguyen et al. [121]**: $f(x) = \alpha\,(P_B(x,y) - P_F(x,y)) + (1-\alpha)\,(1 - 2P(x,y))$, $k(u) = 0$.
- **Spencer-Chen (Convex) [153]**: $f(x) = f_{CV}(x) + \theta\mathcal{D}_E(x)$, $k(u) = \alpha v'_\varepsilon(u)$.

Then if we define a PDE of the general form

$$-\mu\nabla\cdot\left(G(x)\frac{\nabla u}{|\nabla u|_{\varepsilon_2}}\right) + k(u) + f(x) = 0$$

with

(i) Neumann boundary conditions $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ ($\boldsymbol{n}$ the outward normal unit vector)

(ii) $k(u)$ satisfies $k(u) \geq k(v)$ if $u \geq v$

(iii) $k(u)$ and $f(x)$ are bounded; and

(iv) $G(x) = Id$ or $G(x) = f(|\nabla z(x)|) = \frac{1}{1+|\nabla z(x)|^2}$,

we have a unique solution $u \in C([0,T) \times \overline{\Omega})$ for a given initialisation. Consequently we conclude that all above models admit a unique solution.

Proof. The conditions (i)–(iv) hold for all of these models. All of these models require Neumann boundary conditions and use the permitted $G(x)$. The monotonicity of $\nu_\varepsilon'(u)$ is discussed in the proof of (C1) for Theorem 3.4.1.3 and the boundedness of $f(x)$ and $k(u)$ is clear in all cases. □

Remark 3.4.2.2. Theorem 3.4.1.3 and Corollary 3.4.2.1 also generalise to cases where $G(x) = \frac{1}{1+\beta|\nabla z|^2}$ and to $G(x) = \mathcal{D}(x)g(|\nabla z|)$ where $\mathcal{D}(x)$ is a distance function such as in [76, 77, 78, 153]. The proof is very similar to that shown in §3.4.1, relying on Lipschitz continuity of the function $G(x)$.

Remark 3.4.2.3. We cannot apply the classical viscosity solution framework to the Rada-Chen model [135] as this is a non-local problem with

$$k(u) = 2\nu \left(\int_\Omega H_\varepsilon(u) \, \mathrm{d}\Omega - A_1 \right)$$

.

3.5. Numerical Results

In this section we will demonstrate the advantages of the Geodesic Model for selective image segmentation over related and previous models. Specifically we shall compare

- **M1** — the Nguyen et al. (2012) model (2.21) [121];
- **M2** — the Rada-Chen (2013) model (2.22) [135];
- **M3** — the convex Spencer-Chen (2015) model (2.27) [153];
- **M4** — the convex Liu et al. (2018) model (2.28) [110];

- **M5** — the reformulated Rada-Chen model with geodesic distance penalty (see Remark 3.5.0.1);
- **M6** — the reformulated Liu et al. model with geodesic distance penalty (see Remark 3.5.0.1);
- **M7** — the proposed convex Geodesic Model (Algorithm 1).

Remark 3.5.0.1 (A note on **M5** and **M6**). We include **M5 – M6** to test how the geodesic distance penalty term can improve **M2** [135] and **M4** [110]. These were obtained as follows:

- we extend **M2** to **M5** simply by including the geodesic distance function $\mathcal{D}_G(x,u)$ in the functional.
- we extend **M4** to **M6** with a minor reformulation to include data fitting terms. The original functional given by (2.28) is

$$\mu \int_\Omega |\nabla u| \, \mathrm{d}\Omega + \mu_2 \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega + \lambda \int_\Omega \omega^2(\boldsymbol{x}) \, |z-u|^2 \, \mathrm{d}\Omega, \tag{3.22}$$

  which is similar to the Mumford-Shah model §sec:MS. We consider only the piecewise constant reformulation given by

$$\mu \int_\Omega |\nabla \phi| \, \mathrm{d}\Omega + \int_\Omega \omega^2(x,y) \left[\lambda_1 (z-c_1)^2 - \lambda_2 (z-c_2)^2\right] \phi \, \mathrm{d}\Omega, \tag{3.23}$$

  where z is the given image, ϕ is a level set function and c_1, c_2 are the average intensities in the regions where ϕ is positive and negative respectively. This is a non-convex model and can be reformulated using the technique of Chan et al. discussed in §2.2.6.3 including the penalty fitting term $\nu_\varepsilon(\phi)$. Note that we now follow convention and change notation from ϕ to u as the model changes from non-convex to convex. By also including the geodesic distance fitting term, the model **M6** is

$$\begin{aligned} \min_{u,c_1,c_2} \Big\{ \mathcal{F}_{CV\omega}(u,c_1,c_2) = & \int_\Omega \omega^2(x,y) \left[\lambda_1 (z(x,y)-c_1)^2 - \lambda_2 (z(x,y)-c_2)^2\right] u \, \mathrm{d}\Omega \\ & + \mu \int_\Omega g(|\nabla z|)) |\nabla u| \, \mathrm{d}\Omega + \theta \int_\Omega \mathcal{D}_G(x,y) u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu_\varepsilon(u) \, \mathrm{d}\Omega \Big\} \end{aligned} \tag{3.24}$$

where u takes values in $[0,1]$ and the segmentation boundary is given by $\{\boldsymbol{x}|u > \gamma\}$ for almost every $\gamma \in (0,1)$. We have $\mu, \lambda_1, \lambda_2$ non-negative fixed real parameters, α and $\nu_\varepsilon(u)$ as defined in (2.18) and ω as defined for the convex Liu et al. model (2.28). This is a convex model and is the same as the proposed Geodesic Model **M7** but with weighted intensity fitting terms.

Four sets of test results are shown below. In Test 1 we compare models **M1 – M6** to the proposed model **M7** for two images which are hard to segment. The first is a CT scan from which we would like to segment the lower portion of the heart, the second is an MRI scan of a knee and we would like to segment the top of the Tibia. See Figure 3.10 for the test images and the marker sets used in the experiments. In Test 2 we will review the sensitivity of the proposed model to the main parameters. In Test 3 we will give several results achieved by the model using marker and anti-marker sets. In Test 4 we show the initialisation independence and marker independence of the Geodesic Model on real images.

For **M7**, we denote by $\tilde{u}$ the thresholded $u > \tilde{\gamma}$ at some value $\tilde{\gamma} \in (0,1)$ to define the segmented region. Although the threshold can be chosen arbitrarily in $(0,1)$ from the work by [44, Thm 1] and [153], we usually take $\tilde{\gamma} = 0.5$.

Quantitative Comparisons. To measure the quality of a segmentation, we use the Tanimoto Coefficient (TC) (or Jaccard Coefficient [95]) defined by

$$TC(\tilde{u}, GT) = \frac{|\tilde{u} \cap GT|}{|\tilde{u} \cup GT|}$$

where GT is the 'ground truth' segmentation and $\tilde{u}$ is the result from a particular model. This measure takes value one for a segmentation which coincides perfectly with the ground truth and reduces to zero as the quality of the segmentation gets worse. In the other tests, where a ground truth is not available, we use visual plots.

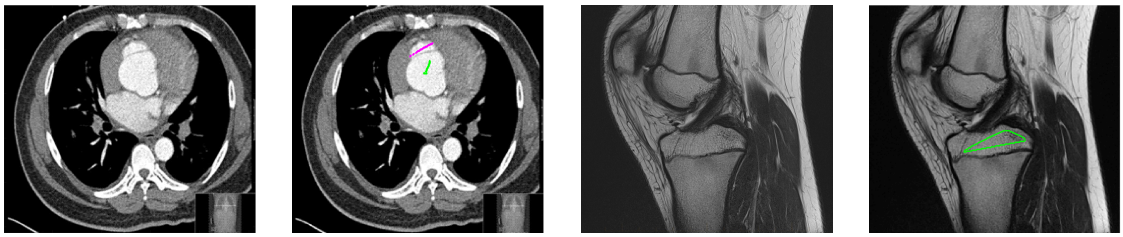
Parameter Choices and Implementation. We set $\mu = 1$, $\tau = 10^{-2}$ and vary $\lambda = \lambda_1 = \lambda_2$ and θ. Following [44] we let $\alpha = ||\lambda_1(z - c_1)^2 - \lambda_2(z - c_2)^2 + \theta \mathcal{D}_G(x,y)||_{L^\infty}$. To implement the marker points in MATLAB we use `roipoly` for choosing a small number of points by clicking and also `freedraw` which allows the user to draw a path of marker points. The stopping criteria used is the dynamic residual falling below a given threshold, i.e. once $||u^{k+1} - u^k||/||u^k|| < tol$ the iterations stop (we use $tol = 10^{-6}$ in the tests shown).

Test 1 – Comparison of models M1 – M7.

In this test we give the segmentation results for models **M1 – M7** for the two challenging test images shown in Figure 3.10. The marker and anti-marker sets used in the experiments are also shown in this figure. After extensive parameter tuning, the best final segmentation results for each of the models are shown in Figures 3.11 and 3.12. For **M1 – M4** we obtain incorrect segmentations in both cases. In particular, the results of **M2** and **M4** are interesting as the former gives poor results for both images, and the latter gives a reasonable result for Test Image 1 and a poor result for Test Image 2. In the case of **M2**, the regularisation term includes the edge detector and the distance penalty term (see (2.22)). It is precisely this which permits the poor result in Figures 3.11(b) and 3.12(b) as the edge detector is zero along the contour and the fitting terms are satisfied there (both intensity and area constraints) – the distance term is not large enough to counteract the effect of these. In the case of **M4**, the distance term and edge detector are separated from the regulariser and are used to weight the Chan-Vese fitting terms (see (2.28)). The poor segmentation in Figure 3.12(b) is due to the Chan-Vese terms encouraging segmentation of bright objects (in this case), weighting ω enforces these terms at all edges in the image and near $\mathcal{M}$. In experiments, we find that **M4** performs well when the object to segment is of approximately the highest or lowest intensity in the image, however when this is not the case, results tend to be poor. We see that, in both cases, models **M5** and **M6** give much-improved results to **M2** and **M4** (obtained by incorporating the geodesic distance penalty into each). The proposed Geodesic Model **M7** gives an accurate segmentation in both cases. It remains to compare **M5, M6** and **M7**. We see that **M5** is a non-convex model (and cannot be made convex [153]), therefore results are initialisation dependent. It also requires one more parameter than **M6** and **M7**, and an accurate set $\mathcal{M}$ to give a reasonable area constraint in (2.22). These limitations lead us to conclude **M6** and **M7** are better choices than **M5**. In the case of **M6**, it has the same number of parameters as **M7** and gives good results. **M6** can be viewed as the model **M7** with weighted intensity fitting terms (compare (3.10) and (3.24)). Experimentally, we find that the same quality of segmentation result can be achieved with both models generally, however **M6** is more parameter sensitive than **M7**. This can be seen in the parameter map in Figure 3.13 with **M7** giving an accurate result for a wider range of parameters than **M6**. To show the improvement of **M7** over previous models, we also give an image in Figure 3.14 which can be accurately segmented with **M7** but the correct

result is never achieved with **M6** (or **M3**). Therefore we find that **M7** outperforms all other models tested **M1 – M6**.

Remark 3.5.0.2. Models **M2 – M7** are coded in MATLAB and use exactly the same marker/anti-marker set. For model **M1**, the software of Nguyen et al. requires marker and anti-marker sets to be input to an interface. These have been drawn as close as possible to match those used in the MATLAB code.

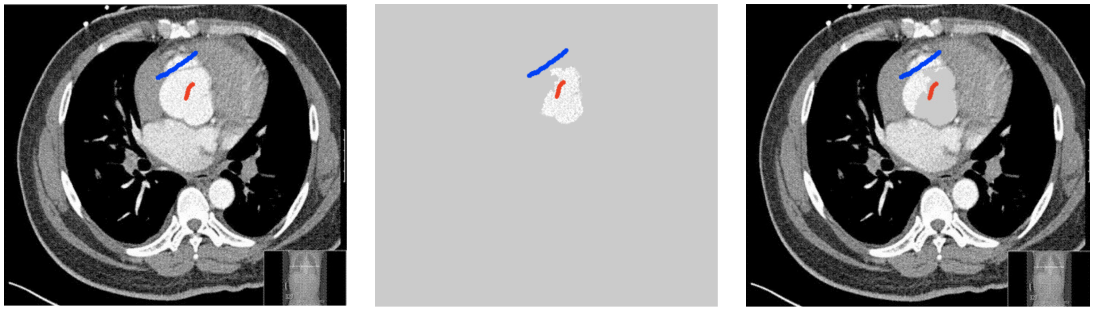


(i) (ii) (iii) (iv)

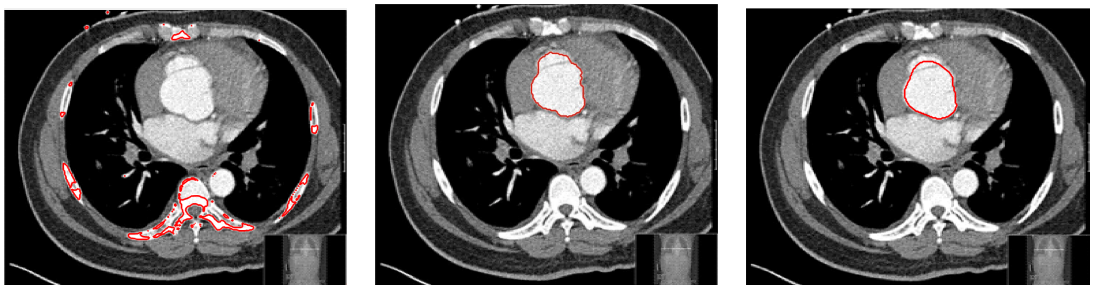
Figure 3.10: Test 1 setting: (i) Image 1; (ii) Image 1 with marker and anti-marker set shown in green and pink respectively; (iii) Test Image 2; (iv) Image 2 with marker set shown.

Test 2 – Test of M7's sensitivity to changes in its main parameters. In this test we demonstrate that the proposed Geodesic Model is robust to changes in the main parameters. The main parameters in (3.12) are $\mu, \lambda_1, \lambda_2, \theta$ and ε_2. In all tests we set $\mu = 1$, which is simply a rescaling of the other parameters, and we set $\lambda = \lambda_1 = \lambda_2$. In the first example, in Figure 3.13, we compare the TC value for various λ and θ values for segmentation of a bone in a knee scan. We see that the segmentation is very good for a larger range of θ and λ values. For the second example, in Figure 3.14, we show an image and marker set for which the Spencer-Chen model (**M3**) and modified Liu et al. model **M6** cannot achieve the desired segmentation for any parameter range, but which can be attained for the Geodesic Model for a vast range of parameters. The final example, in Table 3.1, compares the TC values for various ε_2 values with fixed parameters $\lambda = 2$ and $\theta = 2$. We use the images and ground truth as shown in Figures 3.13 and 3.14: on the synthetic circles image we obtain a perfect segmentation for all values of ε_2 tested, and in the case of the knee segmentation the results are almost identical for any $\varepsilon_2 < 10^{-6}$, above which the quality slowly deteriorates.

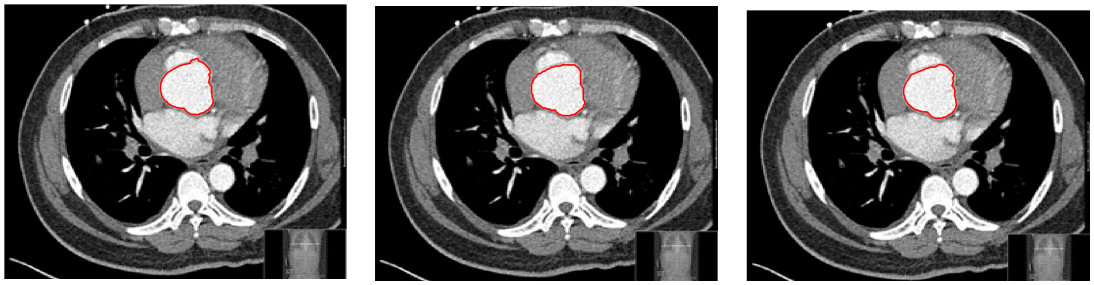
Test 3 – Further Results from the Geodesic Model M7. In this test we give some medical



(a) **M1** (Left to right:) Test Image 1 with markers (red) and anti-markers (blue), foreground segmentation and background segmentation (we used published software, no parameter choice required).



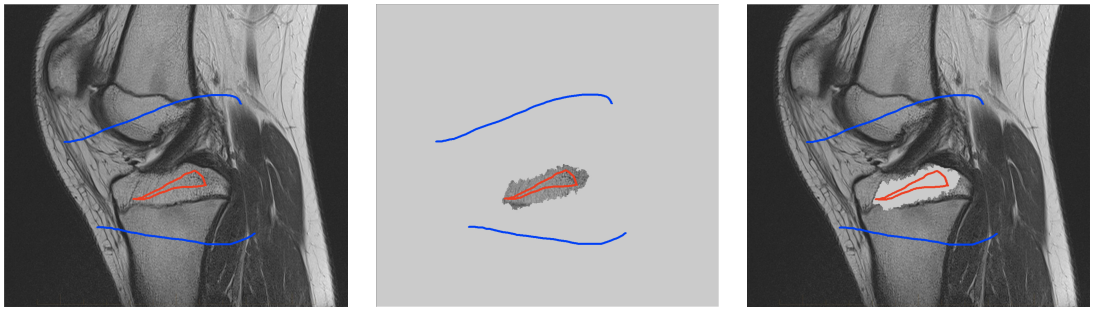
(b) **M2** $\lambda = 1, \gamma = 10.$ (c) **M3** $\lambda = 5, \theta = 3.$ (d) **M4** $\lambda = 1/4.$



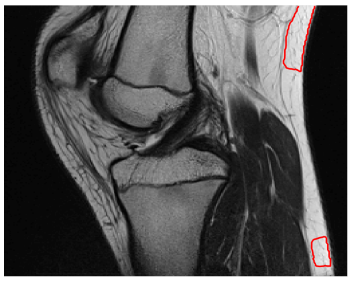
(e) **M5** $\lambda = 5, \gamma = 3, \theta = \frac{1}{10}.$ (f) **M6** $\lambda = 15, \theta = 3.$ (g) **M7** $\lambda = 10, \theta = 1.$

Figure 3.11: Visual comparison of **M1 – M7** results for Test Image 1. **M1** segmented part of the object, **M2 – M4** failed to segment the object, **M5** gave a reasonable result (though not accurate) and, **M6** and **M7** correctly segmented the object.

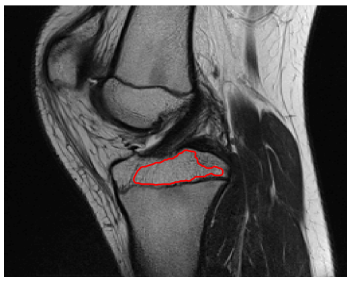
segmentation results obtained using the Geodesic Model **M7**. The results are shown in Figure 3.15. In the final two columns we use anti-markers to demonstrate how to overcome blurred edges and low contrast edges in an image. These are challenging and it is pleasing to see the correctly segmented results.



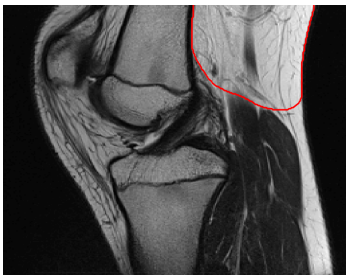
(a) **M1** (Left to right:) Test Image 2 with markers (red) and anti-markers (blue), foreground segmentation and background segmentation (we used published software, no parameter choice required).



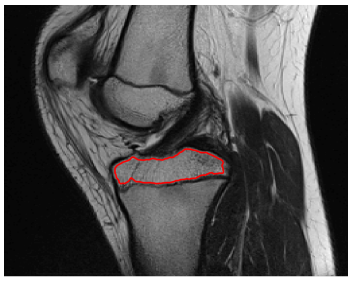
(b) **M2** $\lambda = 1$, $\gamma = 15$.



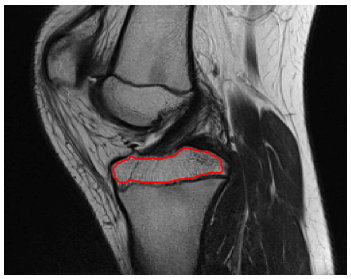
(c) **M3** $\lambda = 5$, $\theta = 1$.



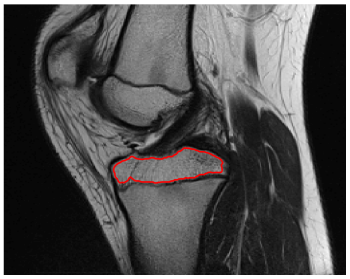
(d) **M4** $\lambda = 1/8$.



(e) **M5** $\lambda = 1, \gamma = 15, \theta = \frac{1}{10}$.



(f) **M6** $\lambda = 15, \theta = 1$.

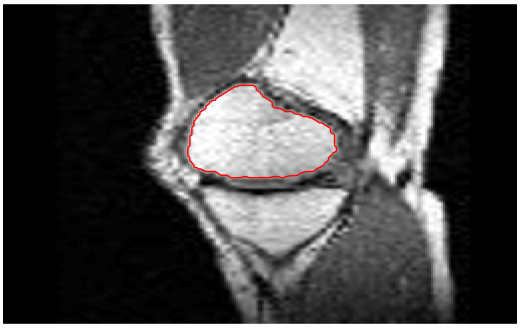


(g) **M7** $\lambda = 10, \theta = 1$.

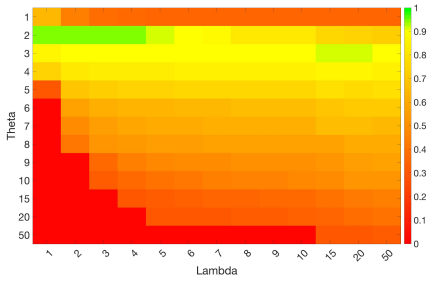
Figure 3.12: Visual comparison of **M1 – M7** results for Test Image 2. **M1** segmented part of the object, **M2 – M4** failed to segment the object, **M5**, **M6** and **M7** correctly segmented the object.

Test 4 – Initialisation and Marker Set Independence. In the first example, in Figure 3.16, we see how the convex Geodesic Model **M7** gives the same segmentation result regardless of initialisation, as expected of a convex model. Hence the model is flexible in implementation. From many experiments it is found that using the polygon formed by

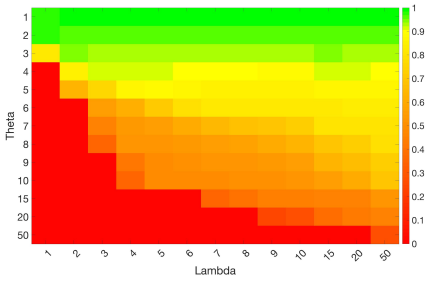
(a) Original Image.



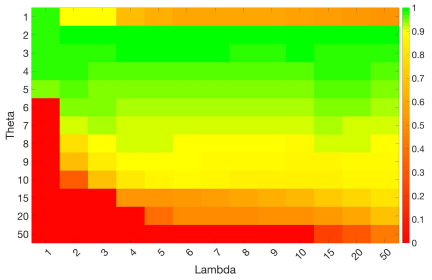
(b) Ground Truth Segmentation.



(c) **M3** TC values for various λ and θ values.



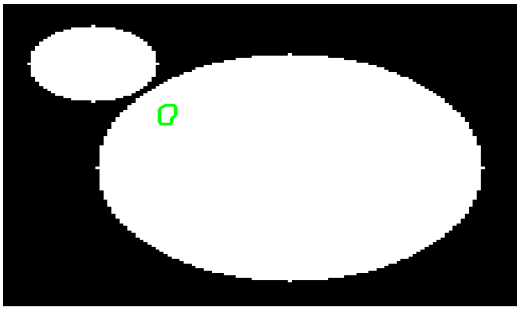
(d) **M6** TC values for various λ and θ values.



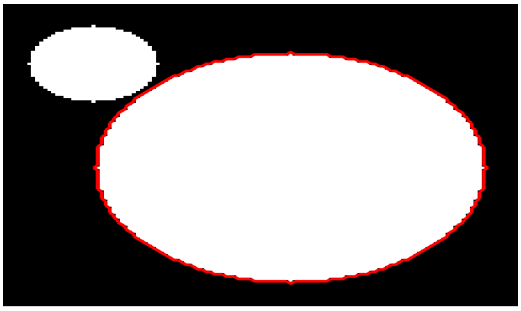
(e) **M7** TC values for various λ and θ values.

Figure 3.13: Parameter heatmaps for **M3**, **M6** and **M7**

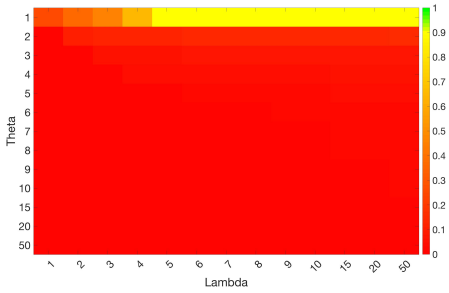
the marker points as the initialisation converges to the final solution faster than using an arbitrary initialisation. In the second example, in Figure 3.17, we show intuitively how Model **M7** is robust to the number of markers and the location of the markers within the object to be segmented. The Euclidean distance term, used in the Spencer-Chen model **M3**, is sensitive to the position and number of marker points, however, regardless of where the markers are chosen, and how many are chosen, the geodesic distance map



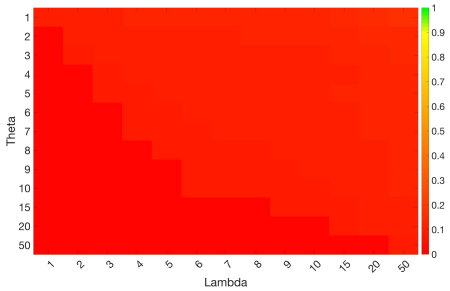
(a) Original image with marker set.



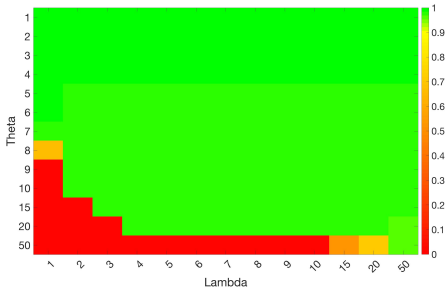
(b) Ground truth segmentation.



(c) **M3** TC values for various λ and θ values.



(d) **M6** TC values for various λ and θ values.



(e) **M7** TC values for various λ and θ values.

Figure 3.14: Parameter heatmaps for **M3**, **M6** and **M7**

will be almost identical.

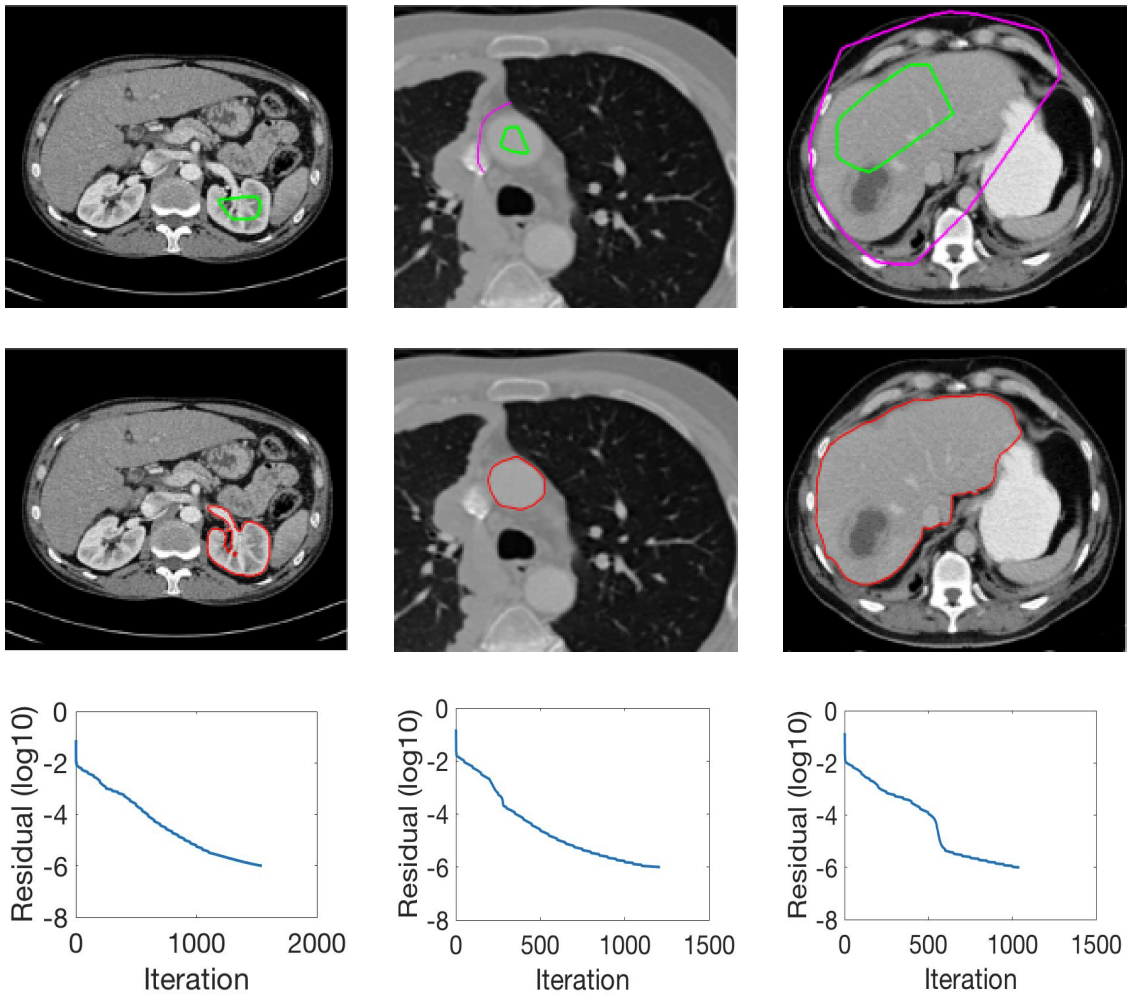


Figure 3.15: Three further test results obtained using our Geodesic Model **M7**, all with parameters $\theta = 5$, $\lambda = 5$. The first row shows the original image with the marker set (plus anti-marker set), the second row the final segmentation result and the final row shows the residual history.

ε_2	Knee Segmentation (Figure 3.13)	Circle Segmentation (Figure 3.14)
10^{-10}	0.97287	1.00000
10^{-8}	0.97287	1.00000
10^{-6}	0.97235	1.00000
10^{-4}	0.96562	1.00000
10^{-2}	0.94463	1.00000
10^{0}	0.90660	1.00000
10^{2}	0.89573	1.00000
10^{4}	0.89159	1.00000

Table 3.1: The Tanimoto Coefficient for various ε_2 values, segmenting the images in Figures 3.13 and 3.14.

(i) (ii) (iii) (iv) (v)

Figure 3.16: Test 4 on **M7**'s initialisations ($\theta = 5, \lambda = 5$). (i) The original image with the marker set indicated; (ii) Initialisation 1 using the image itself; (iii) Segmentation result from Initialisation 1; (iv) Initialisation 2 away from the object to be segmented; (v) Segmentation 2 from initialisation 2. Clearly, **M7** gives the same result.

3.6. Conclusions

In this chapter, a new convex selective segmentation model has been proposed, using geodesic distance as a penalty term. This model gives results that are unachievable by alternative selective segmentation models and is also more robust to the parameter choices. Adaptations to the penalty term have been discussed which make it robust to noisy images and blurry edges whilst also penalising objects far from the marker set (in a Euclidean distance sense). A proof for the existence and uniqueness of the viscosity solution to the PDE given by the Euler-Lagrange equation for the model has been given (which applies to an entire class of image segmentation PDEs). Finally we have con-

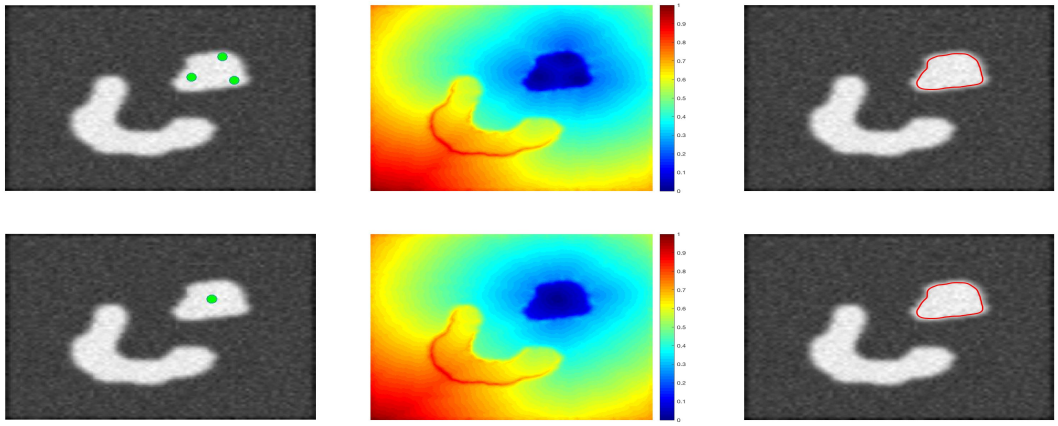


Figure 3.17: Test 4 on **M7**'s marker set ($\theta = 5, \lambda = 3$). Row 1 shows the original image with 3 marker points, the normalised geodesic distance map and the final segmentation result. Row 2 shows the original image with 1 marker point, the normalised geodesic distance map and the final segmentation result. Clearly the second and third columns are the same for different marker points. Thus **M7** is robust.

firmed the advantages of using the geodesic distance with some experimental results. Future works will look for further extension of selective segmentation to other frameworks such as using high-order regularisers [180, 62] where only incomplete theories exist.

Chapter 4

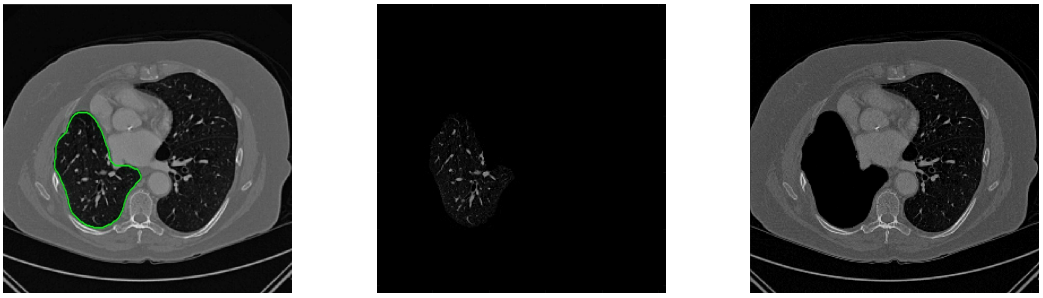
Chan-Vese Reformulation for Selective Image Segmentation

This chapter introduces a reformulation of the Chan-Vese intensity fitting terms for images which have a similar average foreground and background intensity. Typically, in this situation, the Chan-Vese intensity fitting terms do not perform as desired and are close to zero everywhere. We will introduce a new model which not only achieves excellent segmentation results, but is convex and is highly robust to both the main parameters and the user input. The results demonstrate that our model beats a wide array of the current state of the art models for many selective segmentation tasks. This chapter is based on the author's paper [140].

4.1. Introduction

Image segmentation is an important application of image processing techniques in which some, or all, objects in an image are isolated from the background. In other words, for an image $z(\boldsymbol{x}) \in \mathbb{R}^2$, we find the partitioning of the image domain $\Omega \subset \mathbb{R}^2$ into subregions of interest. In the case of two-phase approaches, this consists of the foreground domain Ω_F and background domain Ω_B, such that $\Omega = \Omega_F \cup \Omega_B$. In this work, we concentrate on approaching this problem with variational methods, particularly in cases where user input is incorporated. Specifically, we consider the convex relaxation approach of [31, 41]

and many others. This consists of a binary labelling problem where the aim is to compute a function $u(\boldsymbol{x}) \in \{0,1\}$ indicating regions belonging to Ω_F and Ω_B, respectively. This is obtained by imposing a relaxed constraint on the function, $u \in [0,1]$, and minimising a functional that fits the solution to the data with certain conditions on the regularity of the boundary of the foreground regions.



(a) Image with ground truth. (b) Foreground, $c_1 = 0.15$. (c) Background, $c_2 = 0.19$.

Figure 4.1: CT scan with ground truth segmentation shown (green) and associated intensity values.

We will first introduce the seminal work of Chan and Vese [42], a segmentation model that uses the level set framework of Osher and Sethian [125]. This approach assumes that the image z is approximately piecewise-constant, but is dependent on the initialisation of the level set function as the minimisation problem is non-convex. The Chan-Vese model was reformulated to avoid this by Chan et al. [41], using convex relaxation methods, that has the following data fitting functional

$$f_{CV}(u) = \int_{\Omega} \left(\lambda_1 f_1(\boldsymbol{x}) - \lambda_2 f_2(\boldsymbol{x})\right) u(\boldsymbol{x}) \, \mathrm{d}\Omega, \tag{4.1}$$

where $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ are data fitting terms indicating the foreground and background regions, respectively. In particular, in [42] and [41] these are given by

$$f_1(\boldsymbol{x}) = |z - c_1|^2, \qquad f_2(\boldsymbol{x}) = |z - c_2|^2. \tag{4.2}$$

It should be noted that it is common to fix $\lambda = \lambda_1 = \lambda_2$. The data fitting functional is balanced against a regularisation term. Typically, this penalises the length of the contour. This is represented by the total variation (TV) of the function [42, 144], and is sometimes weighted by an edge detection function $g(s) = 1/(1 + \beta s^2)$ [31, 131, 137, 153]. Therefore,

the regularisation term is given as

$$TV_g(u) := \int_\Omega g(|\nabla z(\boldsymbol{x})|)|\nabla u| \, \mathrm{d}\Omega. \tag{4.3}$$

The convex segmentation problem, assuming fixed constants c_1 and c_2, is then defined by

$$\min_{u\in[0,1]} \left\{ F_{CV}(u, c_1, c_2) = TV_g(u) + f_{CV}(u, c_1, c_2) \right\}. \tag{4.4}$$

In the case where the intensity constants are unknown, it is also possible to minimise F_{CV} alternately with respect to u, c_1, and c_2, however, this would make the problem non-convex and hence dependent on the initialisation of u. Functionals of this type have been widely studied with respect to two-phase segmentation [31, 41, 42], which is our main interest. Alternative choices of data fitting terms can be used when different assumptions are made on the image, z. Examples include [3, 4, 48, 107, 154, 167]. We note that multiphase approaches [33, 162] are also closely related to this formulation although in this chapter we focus on the two-phase problem due to associated applications of interest. It is also important to acknowledge analogous methods in the discrete setting such as [16, 64, 79, 143]. However, we do not go into detail about such methods here, although we introduce the work of [61] in §4.3 and compare corresponding results in §4.7.

In selective segmentation, the idea is to apply additional constraints such that user input is incorporated to isolate specific objects of interest. It is common for the user to input marker points to form a set $\mathcal{M}$, where $\mathcal{M} = \{(x_i, y_i) \in \Omega, 1 \leq i \leq k\}$ and from this we can form a foreground region $\mathcal{P}$ whose interior points are inside the object to be segmented. In the case that $\mathcal{M}$ is provided $\mathcal{P}$ will be a polygon, but any user-defined region in the foreground is consistent with the proposed method. Some examples of selective or interactive methods include [34, 61, 76, 79, 110, 121, 137, 151, 153, 177]. A particular application of this in medical imaging is organ contouring in computed tomography (CT) images. This is often done manually which can be laborious and inefficient and it is often not possible to enhance existing methods with training data. In cases where learning based methods are applicable, the work of Xu et al. [174] and Bernard and Gygli [21] are state of the art approaches. At this stage we define the additional constraints in

selective segmentation as follows:

$$f_S(u) = \theta \int_\Omega \mathcal{D}(\boldsymbol{x}) u \, \mathrm{d}\Omega, \tag{4.5}$$

where $\mathcal{D}(\boldsymbol{x})$ is some distance penalty term, such as [135, 137, 153], and θ is a selection parameter. Essentially, the idea is that the selection term $\mathcal{D}(\boldsymbol{x})$ (based on the region $\mathcal{P}$ formed by the user input marker set) should penalise regions of the background (as defined by the data fitting term $f_2(\boldsymbol{x})$) and also pixels far from $\mathcal{P}$. In this chapter, we choose $\mathcal{D}(\boldsymbol{x})$ to be the geodesic distance penalty proposed in Chapter 3. Explicitly, the geodesic distance from the region $\mathcal{P}$ formed from the marker set is given by:

$$\mathcal{D}_M(\boldsymbol{x}) = 0 \text{ for } \boldsymbol{x} \in \mathcal{P},$$

$$\mathcal{D}_M(\boldsymbol{x}) = \frac{\mathcal{D}^0_M(\boldsymbol{x})}{||\mathcal{D}^0_M(\boldsymbol{x})||_{L^\infty}} \text{ for } \boldsymbol{x} \notin \mathcal{P},$$

where $\mathcal{D}^0_M(\boldsymbol{x})$ is the solution of the following PDE:

$$|\nabla \mathcal{D}^0_M(\boldsymbol{x})| = q(\boldsymbol{x}), \qquad \mathcal{D}^0_M(\boldsymbol{x}_0) = 0, \, (\boldsymbol{x}_0) \in \mathcal{P}. \tag{4.6}$$

The function $q(\boldsymbol{x})$ is image dependent and controls the rate of increase in the distance. It is defined as a function similar to

$$q(\boldsymbol{x}) = \varepsilon_{\mathcal{D}} + \beta_G |\nabla z(\boldsymbol{x})|^2, \tag{4.7}$$

where $\varepsilon_{\mathcal{D}}$ is a small non-zero parameter and β_G is a non-negative tuning parameter. We set the value of $\beta_G = 10^3$ and $\varepsilon_{\mathcal{D}} = 10^{-3}$ throughout. Note that if $q(\boldsymbol{x}) \equiv 1$ then the distance penalty $\mathcal{D}_M(\boldsymbol{x})$ is simply the normalised Euclidean distance, as used in [153].

A general selective segmentation functional, assuming homogeneous target regions, is therefore given by:

$$F_S(u, c_1, c_2) = TV_g(u) + f_{CV}(u, c_1, c_2) + f_S(u). \tag{4.8}$$

Assuming that the optimal intensity constants c_1 and c_2 are fixed, the minimisation

problem is then:

$$\min_{u\in[0,1]} F_S(u, c_1, c_2). \tag{4.9}$$

Again, it is possible to alternately minimise $F_S(u, c_1, c_2)$ with respect to the constants c_1 and c_2 to obtain the average intensity in Ω_F and Ω_B, respectively. However, in selective segmentation, it is often sufficient to fix these according to the user input. In the framework of (4.9) the Chan-Vese terms [41, 42, 119] have limitations due to the dependence on c_2. In conventional two-phase segmentation problems, it makes sense to penalise deviances from c_2 outside the contour, however for selective segmentation we need not consider the intensities outside of the object we have segmented. Regardless of whether the intensity of regions outside the object is above or below c_1, it should be penalised positively. The Chan-Vese terms cannot ensure this as they work based on a fixed "exterior" intensity c_2 and can lead to negative penalties on regions which are outside the object of interest. It is our aim in this chapter to address this problem.

The motivation for this work comes from observing contradictions in using piecewise-constant intensity fitting terms in selective segmentation. Whilst good results are possible with this approach, the exceptional cases lead to severe limitations in practice. This is quite common in medical imaging as demonstrated in Fig. 4.1, where the target foreground has a low intensity. Given that the corresponding background includes large regions of low intensity, the optimal average intensities for this segmentation problem are $c_1 = 0.1534$ and $c_2 = 0.1878$. For cases where $c_1 \approx c_2$, we see that by (4.1), $f_1 - f_2 \approx 0$ almost everywhere in the domain Ω. This means that it is very difficult to achieve an adequate result, without an over-reliance on the user input or parameter selection.

The central premise for applying Chan-Vese type methods is the assumption that the image approximately consists of

$$z(\boldsymbol{x}) = c_1\chi_F + c_2\chi_B + \eta, \tag{4.10}$$

where η is noise, χ_i is the characteristic function of the region Ω_i, for $i = F, B$ respectively. The idea of selective segmentation is to incorporate user input to apply constraints that exclude regions classified as foreground, based on their location in the image. We use a distance constraint which penalises the distance from the user input markers. However, a key problem for selective segmentation is that for cases where the optimal intensity

values c_1 and c_2 are similar, the intensity fitting term will become obsolete as the contour evolves. This is illustrated in Fig. 4.3(b) where we see that the regions inside the lungs and outside the body have a zero contribution from the Chan-Vese fitting terms. The purpose of our approach is to construct a model that is based on assumptions that are consistent with the observed image and any homogeneous target region of interest. A common approach in selective segmentation is to discriminate between objects of a similar intensity [135, 137, 153]. However, the fitting terms in previous formulations [98, 135, 137, 153] aren't applicable in many cases as there are contradictions in the formulation in this context. We will address this in detail in the following section.

In this chapter, our main contribution is to highlight a crucial flaw in the assumptions behind many current selective segmentation approaches and propose a new fitting term in relation to such methods. We demonstrate how our reformulation is capable of achieving superior results and is more robust to parameter choices than existing approaches, allowing for more consistency in practice. In §4.2, we give a brief review of alternative intensity fitting terms proposed in the literature, and detail them in relation to selective segmentation. We then briefly detail alternative selective segmentation approaches to compare our method against in §4.3. In §4.4, we introduce the proposed model, focussing on a fitting term that allows for significant intensity variation in the background domain which leads c_2 to be artificially averaging to a similar value to c_1. In §4.5, we discuss the implementation of each approach in a convex relaxation framework, provide the algorithm in §4.6, and detail some experimental results in §4.7. Finally, in §4.8 we give some concluding remarks.

4.2. Related Approaches

Here, we introduce and discuss work that has introduced alternative data fitting terms closely related to Chan-Vese [42]. In order to make direct comparisons, we convert each approach to the unified framework of convex relaxation [41]. It is worth noting that this alternative implementation is equivalent in some respects, but that the results might differ slightly if using the original methods. We are considering these models in the terms of selective segmentation, so all formulations have the following structure:

$$\min_{u\in[0,1]} \left\{ F(u) = TV_g(u) + f_S(u) + f(u) \right\}. \tag{4.11}$$

where $f_S(u)$ is the modified geodesic distance defined in (4.5) and $f(u)$ is an intensity dependent fitting term. We are interested in the effectiveness of $f(u)$ in this context, which we will focus on next. In particular, we detail various choices of $f(u)$ from the literature that are generalisations of the Chan-Vese approach. In the following, we refer to minimisers of convex formulations, such as (4.11), by u_γ. Here, the minimiser of $F(u)$ is thresholded for $\gamma \in (0, 1)$ in a conventional way [41].

4.2.1. Region-Scalable Fitting (RSF)

The data fitting term from the work of Li et al. [107], known as Region-Scalable Fitting (RSF), consistent with the convex relaxation technique of [41] is given by

$$f_{RSF}(u) = \int_\Omega \left(\lambda_1 f_1(\boldsymbol{x}) - \lambda_2 f_2(\boldsymbol{x})\right) u \, \mathrm{d}\Omega, \tag{4.12}$$

where

$$\begin{aligned} f_1(\boldsymbol{x}) &= \int_\Omega K_\sigma(\boldsymbol{x} - \boldsymbol{y}) \, |z - h_1(\boldsymbol{x})|^2 \, \mathrm{d}\Omega, \\ f_2(\boldsymbol{x}) &= \int_\Omega K_\sigma(\boldsymbol{x} - \boldsymbol{y}) \, |z - h_2(\boldsymbol{x})|^2 \, \mathrm{d}\Omega, \end{aligned} \tag{4.13}$$

and $K_\sigma(\boldsymbol{x})$ is chosen as a Gaussian kernel with scale parameter $\sigma > 0$. The RSF selective formulation is then given as follows:

$$F_{RSF}(u) = TV_g(u) + f_S(u) + f_{RSF}(u). \tag{4.14}$$

The functions $h_1(\boldsymbol{x})$ and $h_2(\boldsymbol{x})$, which are generalisations of c_1 and c_2 from Chan-Vese, are updated iteratively by

$$\begin{aligned} h_1(\boldsymbol{x}) &= \frac{K_\sigma(\boldsymbol{x}) * (u_\gamma \, z)}{K_\sigma(\boldsymbol{x}) * u_\gamma}, \\ h_2(\boldsymbol{x}) &= \frac{K_\sigma(\boldsymbol{x}) * ((1 - u_\gamma) \, z)}{K_\sigma(\boldsymbol{x}) * (1 - u_\gamma)}. \end{aligned} \tag{4.15}$$

Using the RSF fitting term, any deviations of z from h_1 and h_2 are smoothed by the convolution operator, K_σ. This allows for intensity inhomogeneity in the foreground and background of target objects.

4.2.2. Local Chan-Vese (LCV) Fitting

Wang et al. [167] proposed the Local Chan-Vese (LCV) model. In terms of the equivalent convex formulation, the data fitting term is given by

$$f_{LCV}(u) = \int_\Omega \left(f_1(\boldsymbol{x}) - f_2(\boldsymbol{x})\right) u \, \mathrm{d}\Omega \tag{4.16}$$

where

$$\begin{aligned} f_1(\boldsymbol{x}) &= \alpha \left|z - c_1\right|^2 + \beta \left|z^* - z - d_1\right|^2, \\ f_2(\boldsymbol{x}) &= \alpha \left|z - c_2\right|^2 + \beta \left|z^* - z - d_2\right|^2, \end{aligned} \tag{4.17}$$

and $z^* = M_k * z$. Here, M_k is an averaging convolution with $k \times k$ window. The LCV selective formulation is then given as

$$F_{LCV}(u) = \ TV_g(u) + f_S(u) + f_{LCV}(u). \tag{4.18}$$

The values c_1, c_2, d_1, d_2 which minimise this functional for u_γ are given by

$$\begin{aligned} c_1 &= \frac{\int_\Omega z u_\gamma \, \mathrm{d}\Omega}{\int_\Omega u_\gamma \, \mathrm{d}\Omega}, \quad c_2 = \frac{\int_\Omega z(1-u_\gamma) \, \mathrm{d}\Omega}{\int_\Omega (1-u_\gamma) \, \mathrm{d}\Omega}, \\ d_1 &= \frac{\int_\Omega (z^* - z)\, u_\gamma \, \mathrm{d}\Omega}{\int_\Omega u_\gamma \, \mathrm{d}\Omega}, \quad d_2 = \frac{\int_\Omega (z^* - z)\, (1-u_\gamma) \, \mathrm{d}\Omega}{\int_\Omega (1-u_\gamma) \, \mathrm{d}\Omega}. \end{aligned} \tag{4.19}$$

The formulation is minimised iteratively. The LCV fitting term that $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ includes an additional term weighted by the parameters α and β. The principle for the LCV model is that the difference image $z^* - z$ is a higher contrast image than z and a two-phase segmentation on this image can be computed.

4.2.3. Hybrid (HYB) Fitting

Based on extending the LCV model, Ali et al. [3] proposed the following data fitting term,

$$f_{HYB}(u, c_1, c_2, d_1, d_2) = \int_\Omega \left(f_1(\boldsymbol{x}) - f_2(\boldsymbol{x})\right) u \, \mathrm{d}\Omega \tag{4.20}$$

where

$$f_1(\boldsymbol{x}) = \alpha\,|w - c_1|^2 + \beta\,|w^* - w - d_1|^2\,,$$
$$f_2(\boldsymbol{x}) = \alpha\,|w - c_2|^2 + \beta\,|w^* - w - d_2|^2\,. \tag{4.21}$$

Here, $z^* = M_k * z$, $w = z^* z$, and $w^* = M_k * w$, with M_k the averaging convolution as used in the LCV model. The values c_1, c_2, d_1, d_2 are updated in a similar way to [167], with further details found in [3]. The authors refer to this approach as the Hybrid (HYB) Model. The HYB selective formulation is then given as

$$F_{HYB}(u) = TV_g(u) + f_S(u) + f_{HYB}(u). \tag{4.22}$$

The key aim of the HYB model is to account for intensity inhomogeneity in the foreground and background of the image through the product image w. In LCV, the presence of the blurred image z^* in the data fitting term deals with intensity inhomogeneity, whilst including z helps identify contrast between regions. The authors found that the product image $w = z^* z$ can improve the data fitting in both respects. Therefore they construct an LCV-type function with w rather than the original z. Numerical experiments suggest that this approach is more robust.

4.2.4. Generalised Averages (GAV) Fitting

Recently, Ali et al. [4] proposed using the data fitting terms of Chan-Vese in a signed pressure force function framework [177]. They refer to this approach as Generalised Averages (GAV) as they update the intensity constants in an alternative way, detailed below. In the convex framework, we consider the selective GAV functional:

$$F_{GAV}(u) = TV_g(u) + f_S(u) + f_{GAV}(u), \tag{4.23}$$

where $f_{GAV}(u) = f_{CV}(u)$. This is identical to the CV selective formulation (4.8). However, the authors propose an alternative update for the fitting constants c_1 and c_2, given as follows:

$$c_1 = \frac{\int_\Omega z^\beta u_\gamma \,\mathrm{d}\Omega}{\int_\Omega z^{\beta-1} u_\gamma \,\mathrm{d}\Omega}, \qquad c_2 = \frac{\int_\Omega z^\beta (1-u_\gamma) \,\mathrm{d}\Omega}{\int_\Omega z^{\beta-1} (1-u_\gamma) \,\mathrm{d}\Omega}, \tag{4.24}$$

with $\beta \in \mathbb{R}$. If $\beta = 1$, the approach is identical to CV. In [4] the authors assert that the proposed adjustments have the following properties. As $\beta \to \infty$, c_1 and c_2 approach the maximum and minimum intensity in the foreground and background of the image, respectively. Also, as $\beta \to -\infty$, c_1 and c_2 approach the minimum intensity in the foreground and background of the image, respectively. For example, if a high value of β is set, c_1 will take a larger value than in CV which can be useful for selective segmentation. For example, if we consider the image in Fig. 4.1 we can achieve a larger c_2 value by setting $\beta > 1$ and a smaller value by setting $\beta < 1$. Therefore, there is more flexibility when using this data fitting term in selective formulations. However, it should be noted that it involves the selection of the parameter β, which can be difficult to optimise.

4.3. Alternative Selective Segmentation Models

We now introduce two recent methods that incorporate user input to perform selective segmentation. Each involves input in the form of foreground/background regions to indicate relevant structures of interest. An example of this can be seen in Fig. 4.21, where red regions indicate foreground and blue regions indicate background. We compare against the work of Nguyen et al. [121], which uses a similar convex relaxation framework to the proposed approach, and Dong et al. [61], which uses a variation of the random walk approach. We choose these models to compare to as these are state-of-the-art for selective image segmentation and also use the markers and antimarkers we use with our model, they are the most competitive models to ours currently. We summarise the essential aspects of each approach in the following.

4.3.1. Constrained Active Contours (CAC)

The CAC model of Nguyen et al. [121] uses a probability map, $P(\boldsymbol{x})$, from Bai and Sapiro [16] where the geodesic distances from a marker and antimarker set to the foreground/background regions are denoted by $D_F(\boldsymbol{x})$ and $D_B(\boldsymbol{x})$, respectively. An approximation of the probability that a point $\boldsymbol{x}$ belongs to the foreground is then given by

$$P(\boldsymbol{x}) = \frac{D_B(\boldsymbol{x})}{D_F(\boldsymbol{x}) + D_B(\boldsymbol{x})}. \tag{4.25}$$

Foreground/background Gaussian mixture models (GMM) are estimated from the user input. The terms $Pr(\boldsymbol{x}|F)$ and $Pr(\boldsymbol{x}|B)$ denote the probability that a point, $\boldsymbol{x}$, belongs to the foreground and background, respectively. The normalised log-likelihood for each is then given by

$$\begin{aligned} P_F(\boldsymbol{x}) &= -\log Pr(\boldsymbol{x}|F)/(-\log Pr(\boldsymbol{x}|F) - \log Pr(\boldsymbol{x}|B)), \\ P_B(\boldsymbol{x}) &= -\log Pr(\boldsymbol{x}|B)/(-\log Pr(\boldsymbol{x}|F) - \log Pr(\boldsymbol{x}|B)). \end{aligned} \tag{4.26}$$

GMMs are widely used in selective segmentation [16, 61, 64, 79, 143] and the authors in [121] incorporate this idea into the framework we consider with the following data fitting term:

$$h_c(\boldsymbol{x}) = \alpha_0 \left(P_B(\boldsymbol{x}) - P_F(\boldsymbol{x})\right) + (1 - \alpha_0)\left(1 - 2P(\boldsymbol{x})\right), \tag{4.27}$$

for a weighting parameter $\alpha_0 \in [0, 1]$. It is proposed that α_0 is selected automatically as follows:

$$\alpha_0 = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\log Pr(x_i|F) - \log Pr(x_i|B)}{\log Pr(x_i|F) + \log Pr(x_i|B)}\right|, \tag{4.28}$$

where N is the total number of pixels in the image. Defining g_0 as the function $g(s)$ applied to the image $z(\boldsymbol{x})$ and g_p applied to the GMM probability map $P_F(\boldsymbol{x})$, an enhanced edge function is defined as

$$g_c(\boldsymbol{x}) = \beta_0 g_p + (1 - \beta_0) g_0, \tag{4.29}$$

for a weighting parameter $\beta_0 \in [0, 1]$, which can be set automatically in a similar way to (4.28). Thus, Nguyen et al. [121] define the Constrained Active Contours (CAC) Model as

$$\min_{u\in[0,1]} \left\{ \int_\Omega g_c(\boldsymbol{x})|\nabla u(\boldsymbol{x})| \, \mathrm{d}\Omega + \lambda \int_\Omega h_c(\boldsymbol{x}) u(\boldsymbol{x}) \, \mathrm{d}\Omega \right\}. \tag{4.30}$$

They obtain a solution using the split Bregman method of Goldstein et al. [72], although other methods are applicable and will yield similar results. However, that is not the focus of this chapter so we omit the details here. In the results section, §4.7, we will compare our method against CAC to see our data fitting term compares against a GMM-based approach.

4.3.2. Submarkov Random Walks (SRW)

We now introduce a recent selective segmentation method by Dong et al. [61] known as Submarkov Random Walks (SRW). Rather than using the continuous framework of [41], this approach is based in the discrete setting where each pixel in the image is treated as a node in a weighted graph. Random walks (RW) have been widely used for segmentation since the work of Grady [79]. SRW is capable of achieving impressive results with user-defined foreground and background regions. The selective segmentation result can be obtained by assigning a label to each pixel based on the computed probabilities of the random walk approach. For brevity, we do not provide the full details of the method here, however, further details can be found in [61]. We compare SRW to our proposed approach on a CT dataset in §4.7.4.

We now introduce essential notation to understand the approach of [61]. In RW an image is formulated as a weighted undirected graph $G = (V, E)$ with nodes $v \in V$ and edges $e \in E \subseteq V \times V$. Each node v_i represents an image pixel x_i. An edge e_{ij} connects two nodes v_i and v_j and a weight $w_{ij} \in W$ of edge e_{ij} measures the likelihood that a random walker will cross this edge:

$$w_{ij} = \exp\left(-\frac{||I_i - I_j||^2}{\sigma_0}\right) + \epsilon_0, \tag{4.31}$$

where I_i and I_j are pixel intensities, with $\sigma_0, \epsilon_0 \in \mathbb{R}$. In SRW a user indicates foreground/background regions in a similar way to CAC, as shown in Fig. 4.21, and can be viewed as a traditional random walker with added auxiliary nodes. In [61], these are defined as a set of labelled nodes $V_M = \{V^{l_1}, V^{l_2}, ..., V^{l_K}\}$. A set of labels is defined, $LS = \{l_1, l_2, ..., l_K\}$, with K the number of labels $V^{l_k} = \{V_1^{l_1}, V_2^{l_1}, ..., V_{M_K}^{l_K}\}$, and M_k the number of seeds labelled l_k. The prior is then constructed from the seeded nodes (defined by the user). Assuming a label l_k has an intensity distribution H_k (based on GMM learning), a set of auxiliary nodes $H_k = \{h_1, h_2, \cdots, h_K\}$ is added into an expanded graph G_e to define a graph with prior $\bar{G}$. Each prior node is connected with all nodes in V and the weight, w_{ih_k}, of an edge between a prior node h_k and a node $v_i \in V$ is proportional to u_i^k, the probability density belonging to H_k at v_i .

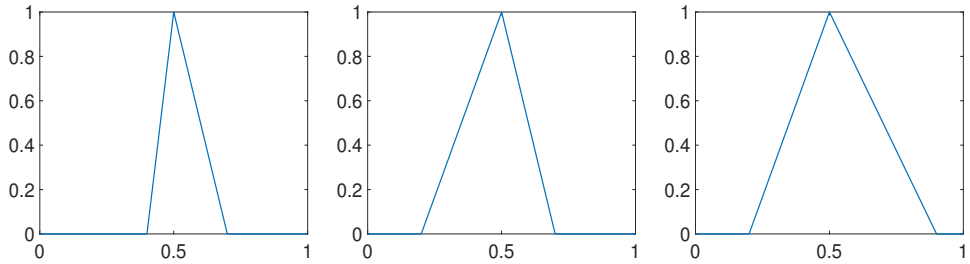
The authors define the probabilities of each node $v_i \in V$ belonging to label l_k as the

average reaching probability, denoted $\bar{r}_i^{l_k}$. This term incorporates the auxiliary nodes introduced above and is dependent on multiple variables and parameters, including w_{ij} (4.31). Further details can be found in [61]. The segmentation result is then found by solving the following discrete optimisation problem:

$$\bar{R}_i = \arg\max_{l_k} \bar{r}_i^{l_k}, \tag{4.32}$$

where $\bar{R}_i$ represents the final label for each node. In other words, for a two-phase segmentation problem, $\bar{R}_i$ is analogous to the discretised solution of a convex relaxation problem in the continuous setting. Comparisons in terms of accuracy can therefore be made directly, which we elaborate on further in §4.7. The authors also detail the optimisation procedure and aspects of dealing with noise reduction.

4.4. Proposed Model



(a) $\gamma_1 = 0.1, \gamma_2 = 0.2$. (b) $\gamma_1 = 0.3, \gamma_2 = 0.2$. (c) $\gamma_1 = 0.3, \gamma_2 = 0.4$.

Figure 4.2: 1D plots of $\tilde{f}_2(\boldsymbol{x})$ for varying γ_1 and γ_2 with $c_1 = 0.5$.

In this section, we introduce the proposed data fitting term for selective segmentation. We consider objects that are approximately homogeneous in the target region. Intrinsically, it is then assumed that the region $\mathcal{P}$, provided by the user, is likely to provide a reasonable approximation of the optimal c_1 value and therefore an appropriate foreground fitting function, f_1, is given by CV (4.2). For this reason, it makes sense to retain this term in the proposed approach. The contradiction is in how the background fitting function f_2 is defined. Considering piecewise-constant assumptions of the image, and many of the related approaches, the background is expected to be defined by a single constant value, c_2. If $c_1 \approx c_2$ then $f_2 \approx f_1$ everywhere, and therefore the fitting term can't

accurately separate background regions from the foreground. It is not practical to rely on $f_S(u)$ to overcome this difficulty as it will produce an over-dependence on the choice of $\mathcal{M}$ and $\mathcal{P}$. This is prohibitive in practice. An alternative function f_2 must therefore be defined which is compatible with f_1 and $f_S(u)$. Here, we define a new data fitting term that penalises background objects in such a way that avoids these problems by allowing intensity variation above and below the value c_1. In order to design a new functional, we first look at the original CV background fitting function

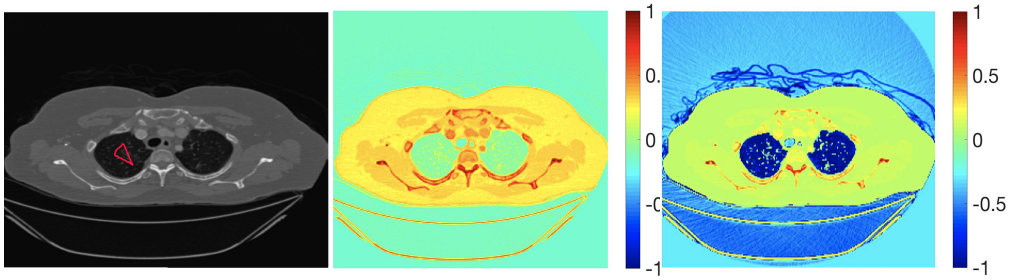
$$f_2 = (z(\boldsymbol{x}) - c_2)^2.$$

It is clear that in an approximately piecewise-constant image this function will be small outside the target region (i.e. where the image takes values near c_2) and positive inside the target region. Our aim in a new fitting term is to mimic this in such a way that is consistent with selective segmentation, where regions with a 'foreground intensity' are forced to be in the background. It is beneficial to introduce two parameters, γ_1 and γ_2, to enforce the penalty on regions of intensity in the range $[c_1 - \gamma_1, c_1 + \gamma_2]$, i.e. enforce the penalty asymmetrically around c_1. We propose the following function to achieve this:

$$\tilde{f}_2(\boldsymbol{x}) = \begin{cases} 1 + \frac{z(\boldsymbol{x}) - c_1}{\gamma_1}, & c_1 - \gamma_1 \leq z(\boldsymbol{x}) \leq c_1 \\ 1 - \frac{z(\boldsymbol{x}) - c_1}{\gamma_2}, & c_1 < z(\boldsymbol{x}) \leq c_1 + \gamma_2 \\ 0, & \textit{else}. \end{cases} \tag{4.33}$$

This function takes its maximum value where $z(\boldsymbol{x}) = c_1$ and is 0 for $z(\boldsymbol{x}) > c_1 - \gamma_1$ and $z(\boldsymbol{x}) < c_1 + \gamma_2$. In Fig. 4.2 we provide a 1D representation of $\tilde{f}_2(\boldsymbol{x})$ for various choices of γ_1 and γ_2, with $z(\boldsymbol{x}) \in [0, 1]$ and $c_1 = 0.5$. Here, it can be seen how the proposed data fitting term acts as a penalty in relation to a fixed constant c_1. It is analogous to CV, whilst accounting for the idea of selective segmentation with a data fitting term. The main advantage of this term is that it replaces the dependence on c_2 in the formulation, which has no meaningful relation to the solution of a selective segmentation problem. Even when the foreground is relatively homogeneous, the background may have intensities of a similar value to c_1 which will cause difficulties in obtaining an accurate solution. We detail the proposed fitting term in the following section.

4.4.1. New Fitting Term



(a) Image and $\mathcal{P}$. (b) CV fitting. (c) New fitting.

Figure 4.3: An image with user input $\mathcal{P}$ shown in red ($c_1 = 0.152, c_2 = 0.188$). Here, we show the difference between the CV fitting function and the proposed approach. The target region is clearly defined by negative values in (iii).

We define the proposed data fitting functional as follows:

$$f_{PM}(u) := \int_{\Omega} (\lambda_1 f_1(\boldsymbol{x}) - \lambda_2 \tilde{f}_2(\boldsymbol{x}))u \ \ \mathrm{d}\Omega, \tag{4.34}$$

for $f_1(\boldsymbol{x}) = (z - c_1)^2$ and $\tilde{f}_2(\boldsymbol{x})$ as defined in (4.33). This is consistent with respect to the intensities of the observed object and the concept of selective segmentation. In Fig. 4.3 we see the difference between CV and the proposed fitting terms for given user input on a CT image. For the CT image, the CV fitting terms are near 0 within the target region and hence the CV terms are simply not contributing at these pixels, this is despite there being a distinct homogeneous area with good contrast on the boundary. Therefore only the distance term implements a penalty at these pixels which is not what we would like, we desire intensity and distance contributions. This illustrates the problem we are aiming to overcome. With the proposed fitting term this phenomenon should be avoided in cases like this. By defining $\tilde{f}_2$ as in (4.33) there is no contradiction if the foreground and background intensities of the target region are similar.

For images where we assume that the target foreground is approximately homogeneous, we have generally found that fixing c_1 according to the user input is preferable. We compute c_1 as the average intensity inside the region $\mathcal{P}$ formed from the user input marker point set. We therefore propose to minimise the following functional with respect

to $u \in [0,1]$, given a fixed c_1 :

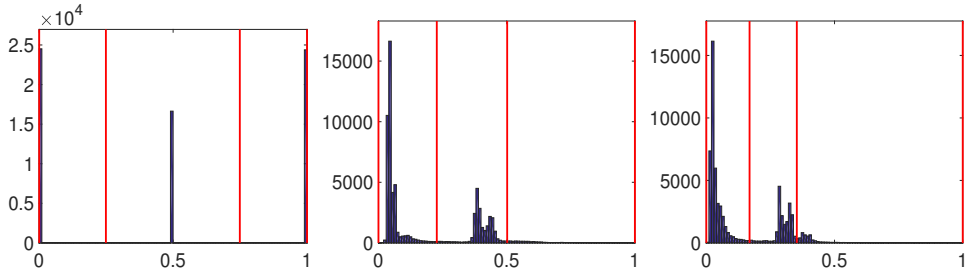
$$F_{PM}(u) = TV_g(u) + f_{PM}(u) + f_S(u). \tag{4.35}$$

where f_S is the geodesic distance computed as described earlier using (4.6). The minimisation problem is given as

$$\min_{u\in[0,1]} F_{PM}(u) \tag{4.36}$$

The model consists of weighted TV regularisation with a geodesic distance constraint as in Chapter 3. However, alternative constraints are possible, such as Euclidean [153], or moments [98]. It is important to note that we have defined the model in a similar framework to the related approaches discussed previously. The main idea is to establish how the proposed fitting term, $f_{PM}(u)$, performs compared to alternative methods. Next we describe how we determine the values of γ_1 and γ_2 in the function $\tilde{f}_2(\boldsymbol{x})$ automatically. This is important in practice as it avoids any additional user input or parameter dependence to achieve an accurate result. In subsequent sections we provide details of how we obtain a solution for the proposed model.

4.4.2. Parameter Selection



(a) Test Image 1. (b) Test Image 7. (c) Test Image 9.

Figure 4.4: The histogram of image intensities for the images referenced. The red lines are the automatic thresholds T_i obtained by Otsu's thresholding with $N = 3$.

For a particular problem it is quite straightforward to optimise the choice of γ_1 and γ_2 experimentally, but we would like a method which is not sensitive to the choice of γ_1 and γ_2 and would also prefer that the user need not choose these values manually. Therefore, in this section we explain how to choose these values automatically based on justifiable

assumptions about general selective segmentation problems. To select the parameters γ_1 and γ_2 we use Otsu's method [126] to divide the histogram of image intensities into N partitions. Otsu's thresholding is an automatic clustering method which chooses optimal threshold values to minimise the intra-class variance. This has been implemented very efficiently in MATLAB in the function `multithresh` for dividing a histogram such that there are $N-1$ thresholds T_i.

We use the thresholds from Otsu's method to find γ_1 and γ_2 as follows. There are three cases to consider, based on the value of c_1 computed from the user input: i) $T_{i-1} \leq c_1 \leq T_i$ for some $i > 1$, ii) $0 \leq c_1 \leq T_1$, iii) $T_{N-1} \leq c_1 \leq 1$. For each case we set the parameters as follows:

(i) $\gamma_1 = c_1 - T_{i-1}, \quad \gamma_2 = T_i - c_1$

(ii) $\gamma_1 = c_1, \quad \gamma_2 = T_1 - c_1$

(iii) $\gamma_1 = c_1 - T_{N-1}, \quad \gamma_2 = 1 - c_1$

Choosing N too large could mean γ_1 and γ_2 are too small as the histogram would be partitioned too precisely. Generally we only ever need to consider a maximum of 3 phases for selective segmentation. If there is a large number of pixels in the image with intensity above or below c_1 the image can be considered two-phase in practice. Conversely, if a large number of pixels in the image have an intensity above and below c_1 the image can essentially be considered three-phase in the context of selective segmentation. This is due to the way $\tilde{f}_2$ has been defined. Therefore, we set $N = 3$ for all tests. In Fig. 4.4 we can see the Otsu thresholds chosen for various images given in this chapter. They divide the peaks in the histogram well and once we know the value of c_1 (the approximation of the intensity of the object we would like to segment) we can automatically choose γ_1 and γ_2 according to this criteria.

4.5. Numerical Implementation

We now introduce the framework in which we compute a solution to the minimisation of the proposed model, as well the related models introduced in Sections 4.1 and 4.2. All

consist of the minimisation problem

$$\min_{u\in[0,1]} \left\{ F_X(u) = TV_g(u) + f_X(u) + f_S(u) \right\}, \tag{4.37}$$

for $X =$ CV, RSF, LCV, HYB, GAV, PM respectively. Minimisation problems of this type (4.37) have been widely studied in terms of continuous optimisation in imaging, including two-phase segmentation. A summary of such methods in recent years is given by Chambolle and Pock [40]. Our numerical scheme follows the original approach in [41]: enforcing the constraint in (4.37) with a penalty function, and deriving the Euler-Lagrange of the regularised functional. We then solve the corresponding PDE by following a splitting scheme first applied to this kind of problem by Spencer and Chen [153]. Whilst the numerical details are not the focus of the work, it is important to note widely used alternative methods. It has proved very effective to exploit the duality in the functional and avoid smoothing the TV term. A prominent example is the split Bregman approach for segmentation by Goldstein et al. [72]. This is closely related to augmented Lagrangian methods, a matter further discussed by Boyd et al. [25]. Analogous approaches also consist of the first-order primal-dual algorithm of Chambolle and Pock [39] and the max-flow/min-cut framework detailed by Yuan et al. [175]. There are practical advantages in implementing such a numerical scheme for our problem, primarily in terms of computational speed. However, in the numerical tests, we include we're mainly interested in accuracy comparisons. For this purpose the convex splitting algorithm of [153] is sufficient, and the extension of splitting schemes for convex segmentation problems may be of interest. Further details can be found in [153] and [137]. In the following, we first discuss the minimisation of (4.37) in a general sense and then mention some important aspects in relation to the alternative fitting terms discussed in §4.2.

4.5.1. Finding the Global Minimiser

To solve this constrained convex minimisation problem (4.38) we use the Additive Operator Splitting (AOS) scheme from Gordeziani et al. [75], Lu et al. [113] and Weickert et al. [170]. This is used extensively for image segmentation models [135, 137, 153]. It allows the 2D problem to be split into two 1D problems, each solved separately, with the results combined in an efficient manner. We address some aspects of AOS in §4.6, with further details provided in [137, 153].

A challenge with the functional (4.35), particularly with respect to AOS, is that this is a constrained minimisation problem. Consequently, it is reformulated by introducing an exact penalty function, $\nu(u)$, given in [41]. To simplify the formulation we define

$$r(\boldsymbol{x}) = \theta\mathcal{D}(\boldsymbol{x}) + f(\boldsymbol{x}),$$

$f(\boldsymbol{x})$ is the function associated with $f_X(u)$. We introduce a new parameter, $\tilde{\lambda}$, which allows us to balance the data fitting terms to the regularisation term more reliably. To be clear, we still only have two main tuning parameters (θ and $\tilde{\lambda}$) as we fix any variable parameters in $f(\boldsymbol{x})$ according to the choices in the corresponding papers. The unconstrained minimisation problem is then given as:

$$\min_u \left\{ TV_g(u) + \tilde{\lambda} \int_\Omega r(\boldsymbol{x})u \, \mathrm{d}\Omega + \alpha \int_\Omega \nu(u) \, \mathrm{d}\Omega \right\}. \tag{4.38}$$

We rescale the data term with $\mathcal{F}(\boldsymbol{x}) = r(\boldsymbol{x})/||r(\boldsymbol{x})||_\infty$. In effect, this change is simply a rescaling of the parameters. This allows for the parameter choices between different models to be more consistent, as the fitting terms are similar in value. The problem (4.38) has the corresponding Euler-Lagrange equation (for fixed c_1):

$$\nabla \cdot \left(g(|\nabla z|) \frac{\nabla u}{|\nabla u|_{\varepsilon_1}} \right) - \tilde{\lambda}\mathcal{F}(\boldsymbol{x}) - \alpha \nu'_{\varepsilon_2}(u) = 0. \tag{4.39}$$

in Ω and $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ where $\boldsymbol{n}$ is the outward unit normal. The constraint is enforced for $\alpha > \frac{\tilde{\lambda}}{2}||r(\boldsymbol{x})||$ by [41]. Two parameters, ε_1 and ε_2, are introduced here. The former is to avoid singularities in the TV term and the latter is associated with the regularised penalty function $\nu_{\varepsilon_2}(u)$ from [153]:

$$\nu_{\varepsilon_2}(u) = H_{\varepsilon_2}\left(b_{\varepsilon_2}(u)\right)\left[b_{\varepsilon_2}(u)\right], \tag{4.40}$$

with $b_{\varepsilon_2}(u) = \sqrt{(2u-1)^2 + \varepsilon_2} - 1$ and regularised Heaviside function

$$H_{\varepsilon_2}(u) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{u}{\varepsilon_2}\right)\right). \tag{4.41}$$

The viscosity solution of the parabolic formulation of (4.39), obtained by multiplying the PDE by $|\nabla u|$, exists and is unique. The general proof for a class of PDEs to which

(4.39) belongs, is included in [137] and we refer the reader there for the details. Once the solution to (4.39) is found, denoted u^*, we define the computed foreground region as follows:

$$u_\gamma = \{\boldsymbol{x} \in \Omega |\ u^*(\boldsymbol{x}) > \gamma\}. \tag{4.42}$$

We select $\gamma = 0.5$ (although other values $\gamma \in (0, 1)$ would yield a similar result according to Chan et al. [41]). In the following, we use the binary form of the solution, u^*, denoted u_γ. This partitions the domain into Ω_F and Ω_B according to the labelling function u_γ.

4.5.2. Implementation for Related Models

The discussion in this section so far has used the function $f(\boldsymbol{x})$ associated with the data fitting functional $f_X(u)$. These corresponding equations for the RSF, LCV, HYB and GAV models are detailed in §4.2, CV is discussed in §4.1, and our approach is given by eqn. (4.34). We use this implementation to obtain selective segmentation versions of each of those models, given by (4.37). When these terms contain parameter choices we follow the advice in the corresponding papers, as far as possible, unless we have found that alternatives will improve results. In the next section, we will give the results of these models and compare them to our proposed approach.

Note. We now discuss details behind tuning parameters for the GAV model. It is noted in §4.2 that the GAV model requires a parameter β to adapt the c_1 and c_2 calculation. We find that it is actually better to consider c_1 and c_2 separately to achieve improved results, as sometimes we wish to tune the values to have a higher c_1 and lower c_2 (or vice-versa) simultaneously. Therefore we introduce parameters β_1 and β_2 to tune c_1 and c_2 as follows:

$$c_1 = \frac{\int_\Omega z^{\beta_1} u}{\int_\Omega z^{\beta_1 - 1} u}\, \mathrm{d}\Omega, \qquad c_2 = \frac{\int_\Omega z^{\beta_2} (1-u)}{\int_\Omega z^{\beta_2 - 1} (1-u)}\, \mathrm{d}\Omega, \tag{4.43}$$

In all experiments, we tested the following combinations of (β_1, β_2): $(1.5, 0.5)$, $(2, 0)$, $(3, -1)$, $(4, -2)$, $(0.5, 1.5)$, $(0, 2)$, $(-1, 3)$ and $(-2, 4)$. For each choice, we optimised the values of $\tilde{\lambda}$ and θ according to the procedure described in §4.7.1. This allowed us to select the optimal combination of (β_1, β_2) for each image.

4.6. Algorithm

Here, we will discuss the algorithm that we use to minimise the selective segmentation model (4.37). We utilise additive operator splitting techniques to solve the minimisation problem efficiently.

4.6.1. An Additive Operator Splitting (AOS) Scheme

Additive Operator Splitting (AOS) [75, 113, 170] is a widely used method for solving PDEs with linear and non-linear diffusion terms [135, 137, 153] such as

$$\frac{\partial u}{\partial t} = \mu \nabla \cdot (G(u)\nabla u) - f_0. \tag{4.44}$$

AOS allows us to split the two-dimensional problem into two one-dimensional problems, which we solve separately and then combine. Each one-dimensional problem gives rise to a tridiagonal system of equations which can be solved efficiently by Thomas' algorithm, hence AOS is a very efficient method for solving PDEs of this type. AOS is a semi-implicit method and permits far larger time-steps than the corresponding explicit schemes would. Hence AOS is more stable than an explicit method [170]. Note here that

$$G(u) = \frac{g(|\nabla z|)}{|\nabla u|_{\varepsilon_1}}, \qquad f_0 = \tilde{\lambda}\mathcal{F}(\boldsymbol{x}) + \alpha \nu'_{\varepsilon_2}(u), \tag{4.45}$$

and $\mu = 1$. The standard AOS scheme assumes f_0 does not depend on u, however in this instance that is not the case. This requires a modification to be used for convex segmentation problems, first introduced by [153]. This non-standard formulation incorporates the regularised penalty term, $\nu_{\varepsilon_2}(u)$, into the AOS scheme which we briefly detail next.

The authors consider the Taylor expansions of $\nu'_{\varepsilon_2}(u)$ around $u = 0$ and $u = 1$. They find that the coefficient b of the linear term in u is the same for both expansions. Therefore, for a change in u of δu around $u = 0$ and $u = 1$ the change in $\nu'_{\varepsilon_2}(u)$ can be approximated by $b \cdot \delta(u)$. To address this, the relevant interval is defined as

$$I_\zeta := [0 - \zeta, 0 + \zeta] \cup [1 - \zeta, 1 + \zeta]$$

and a corresponding update function is given as

$$\tilde{b}(\boldsymbol{x}) = \begin{cases} b, & x \in \Omega, \;\; u(\boldsymbol{x}) \in I_\zeta \\ 0, & else. \end{cases}$$

The solution for (4.44) is then obtained by discretising the equation as follows:

$$\frac{u^{(k+1)} - u^{(k)}}{\tau} = \mu \sum_{\ell=1,2} A_\ell(u^{(k)}) u^{(k+1)} + \alpha \tilde{b}^{(k)}(u^{(k)} - u^{(k+1)}) - f_0^{(k)}.$$

where A_1 and A_2 are discrete forms of $\partial_x(G(u)\partial_x)$ and $\partial_y(G(u)\partial_y)$, respectively (given in [137, 153]). The modified AOS update is then given by

$$u^{(k+1)} = \frac{1}{2} \sum_{\ell=1}^{2} \left(I - 2\tau\mu(I + \tilde{B}^{(k)})^{-1} A_\ell(u^{(k)}) \right)^{-1} \tilde{u}^{(k)}, \tag{4.46}$$

where $\tilde{B}^{(k)} = \text{diag}(\tau\alpha\tilde{b}^{(k)})$ and $\tilde{u}^{(k)} = u^{(k)} + \tau(I + \tilde{B}^{(k)})^{-1} f_0^{(k)}$. This scheme allows for more control on the changes in f_0 between iterations due to the function $\tilde{b}$ and parameter ζ, and therefore leads to a more stable convergence. We refer the reader to [153] for full details of the numerical method.

4.6.2. The Proposed Algorithm

In Algorithm 1 we provide details of how we find the minimiser of the various selective segmentation models detailed above, defined by (4.37). The algorithm is in a general form to be applied to any of the approaches discussed so far. It is important to reiterate that alternative solvers to AOS are available, such as the dual formulation [8, 31, 38], split-Bregman [72], augmented Lagrange [23], primal-dual [39], and max-flow/min-cut [175]. In all experiments, we use the tolerance of 10^{-4} for the stopping criteria and set $\varepsilon_1 = 10^{-4}$, $\varepsilon_2 = 10^{-1}$ and $\tau = 10^{-2}$.

Algorithm 1: Selective Segmentation Algorithm

Provide user input region $\mathcal{P}$ and compute $\mathcal{D}$, according to (4.6).
Define $f(\boldsymbol{x})$ appropriately for the model (CV, RSF, LCV, HYB, GAV, or the proposed approach).
Compute $r(\boldsymbol{x}) = \theta\mathcal{D}(\boldsymbol{x}) + f(\boldsymbol{x})$. and $\mathcal{F}(\boldsymbol{x}) = r(\boldsymbol{x})/||r(\boldsymbol{x})||_\infty$.
Initialise u (arbitrary for $u \in [0,1]$).
while $\delta > tolerance$ **do**
  $u_{old} := u$.
  Update u according to the AOS iteration (4.46).
  $\delta = ||u - u_{old}||/||u_{old}||$.
end while
return $u^* = u$ and binary labelling function, u_γ.

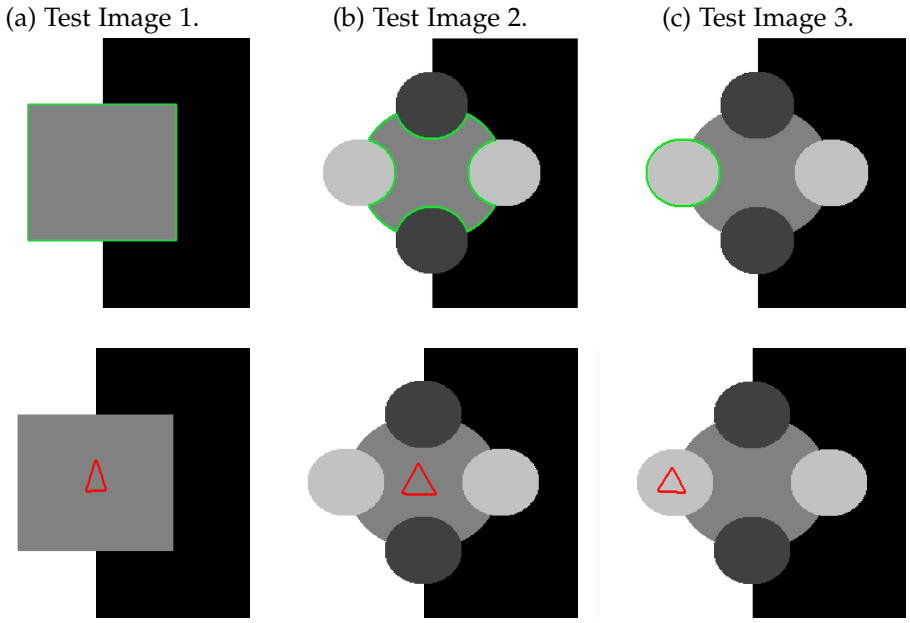


Figure 4.5: Test Images 1–3; the ground truth contours are defined in the first row and the corresponding user input marker set is shown in the second row. These are synthetic images with homogeneous foregrounds selected to highlight the requirement of the new model.

4.7. Results

In this section, we will present results obtained using the proposed model and compare them to using fitting terms from similar models (CV [42], RSF [107], LCV [167], HYB

(a) Test Image 4. (b) Test Image 5. (c) Test Image 6.

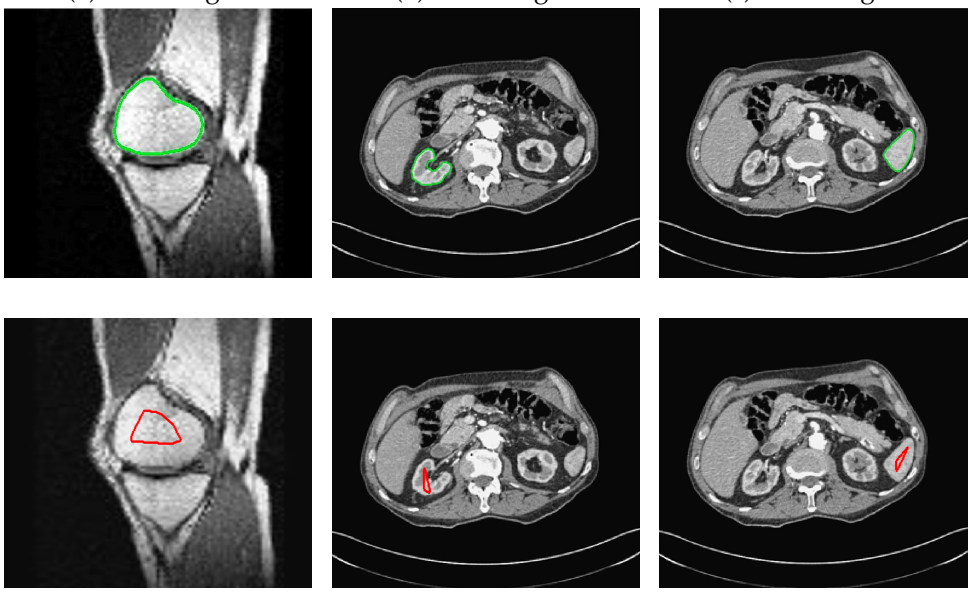


Figure 4.6: Test Images 4–6; the ground truth contours are defined in the first row and the corresponding user input marker set is shown in the second row. These are real images with some degree of intensity inhomogeneity in the foreground, with potential medical applications.

[3], GAV [4]), detailed in §4.2. We intend to provide an overview of how effective each model is in a number of key respects and analyse their potential for practical use in a reliable and consistent manner. Our focus is on how each model can be extended to a consistent selective segmentation framework. The key questions we consider are:

(i) How sensitive are the results to variations of the parameters $\tilde{\lambda}$ and θ?

(ii) Is the model capable of achieving accurate results?

(iii) To what extent is the proposed model dependent on the user input?

Test Images. We will perform all tests on the images shown in Figs. 4.5–4.7. We have provided the ground truth and initialisation used for each image. Test Images 1–3 are synthetic, Test Image 4 is an MRI scan of a knee, Test Images 5–6 are abdominal CT scans, and Test Images 7–9 are lung CT scans. They have been selected to present challenges relevant to the discussion in §4.2. We focus on medical images as this is the application

(a) Test Image 7. (b) Test Image 8. (c) Test Image 9.

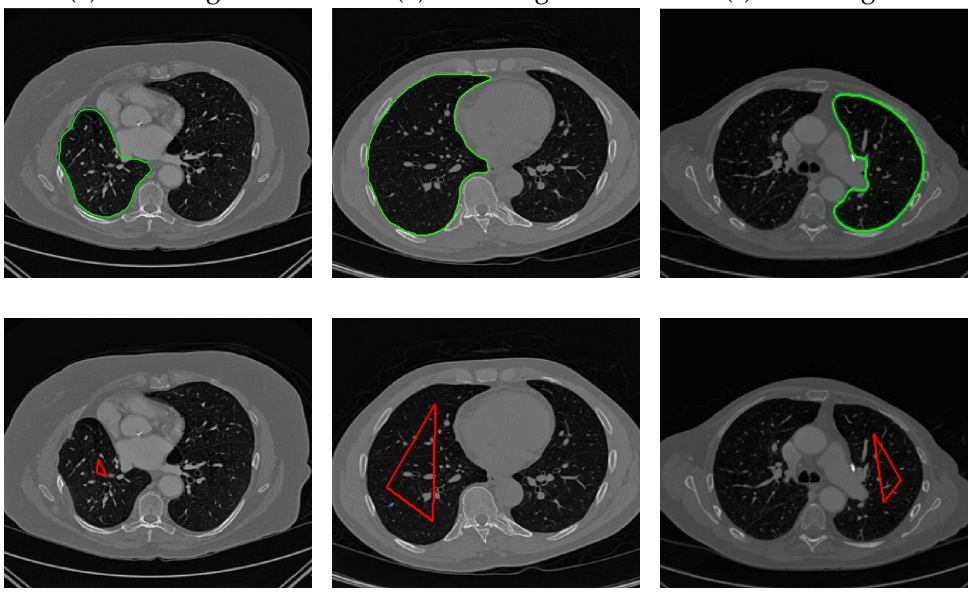


Figure 4.7: Test Images 7–9; the ground truth contours are defined in the first row and the corresponding user input marker set is shown in the second row. These are real images with approximately homogeneous foregrounds. The challenge is that the background contains substantial regions of similar intensity.

of most interest to our work. In the following, we will discuss the results in terms of synthetic images (1–3) and real images (4–9).

Measuring Segmentation Accuracy. In our tests we use the Jaccard Coefficient [95], often referred to as the Tanimoto Coefficient (TC), to measure the quality of the segmentation. This was introduced earlier in §3.5 but we repeat it here for completion. We define accuracy with respect to a ground truth, GT, given by a manual segmentation:

$$GT = \{\boldsymbol{x} \in \Omega | \ \boldsymbol{x} \in \text{foreground}\}.$$

The Tanimoto Coefficient is then calculated as

$$\text{TC} = \frac{|N(u_\gamma \cap GT)|}{|N(u_\gamma \cup GT)|},$$

Figure 4.8: Colour scaling corresponding to TC values, representing the accuracy of the result. This scale is used in subsequent figures.

where $N(\cdot)$ refers to the number of points in the enclosed region. This takes values in the range $[0,1]$, with higher TC values indicating a more accurate segmentation. In the following, we will represent accuracy visually from red (TC $= 0$) to green (TC $= 1$), with the intermediate scaling of colours used shown in Fig. 4.8. This will be particularly relevant in §4.7.2.

Note. In §4.2.4 we mentioned the tuning of parameters in the GAV model. To be explicit the optimal (β_1, β_2) pairs used in the following tests were (4,-2) for Test Images 1 and 2, (1.5,0.5) for Test Images 3,4, and 6, (2,0) for Test Image 5, and (-2,4) for Test Images 7,8, and 9. Results vary significantly as (β_1, β_2) are varied, but we found these to be the best choices for each image.

The discussion of results is split into four sections, addressing the questions introduced above. First, in Section 4.7.1, we will examine the robustness to the parameters $\tilde{\lambda}$ and θ for each model. Then, in Section 4.7.2, we will compare the optimal accuracy achieved by each method to determine what each data fitting term is capable of in the context of selective segmentation for these examples. In Section 4.7.3, we will test the proposed model with respect to the user input. By randomising the input we will determine to what extent the proposed model is suitable for use in practice. Finally, we will compare the proposed approach to the methods introduced in §4.3 on an additional CT dataset. This will help establish how the algorithm performs against competitive approaches in the literature.

4.7.1. Parameter Robustness

In these tests, we aim to demonstrate how sensitive to parameter choices each choice of fitting term is. To accomplish this we perform the segmentations for each of the models discussed (CV, RSF, LCV, HYB, GAV) and the proposed model for a wide range of parameters and compute the TC value. The parameter range used is $\tilde{\lambda}, \theta \in [1, 50]$. Due

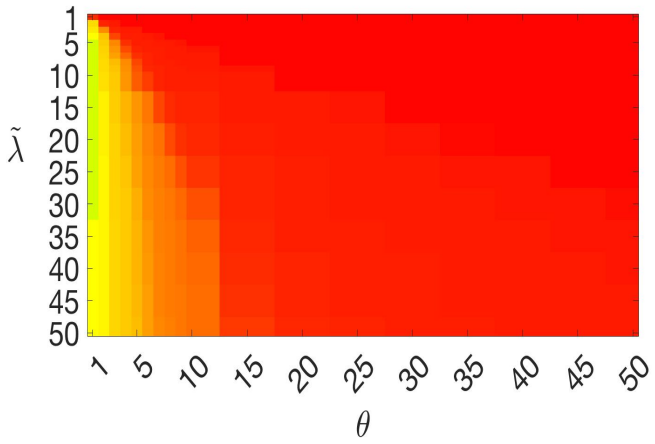


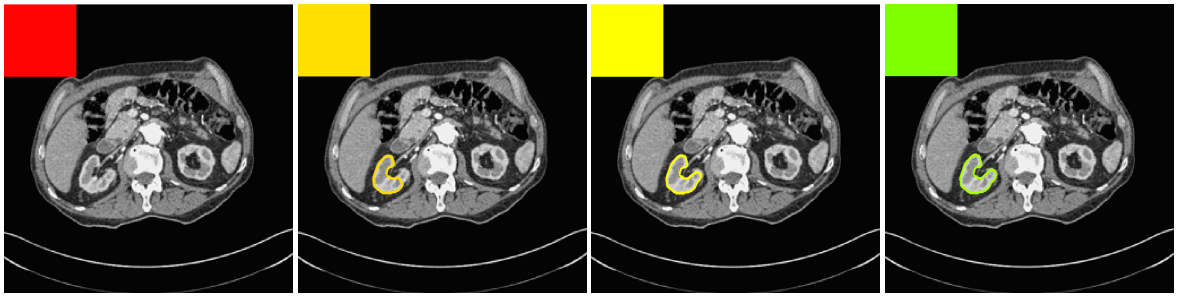
Figure 4.9: Example heatmap of TC values to display segmentation accuracy for parameters $(\tilde{\lambda}, \theta)$.

Model	Test Image								
	1	2	3	4	5	6	7	8	9
CV	0.000	0.000	0.970	0.969	0.933	0.988	0.889	0.931	0.180
RSF	**1.000**	0.997	0.993	0.924	0.884	0.956	0.785	0.950	0.782
LCV	0.313	0.142	0.970	0.970	0.941	0.988	0.911	0.960	0.828
HYB	0.184	0.091	0.988	0.960	0.870	0.988	0.000	0.000	0.000
GAV	0.984	0.960	0.988	0.967	0.965	0.988	0.950	0.954	0.919
Proposed	**1.000**	**1.000**	**1.000**	**0.973**	**0.989**	**0.990**	**0.965**	**0.961**	**0.971**

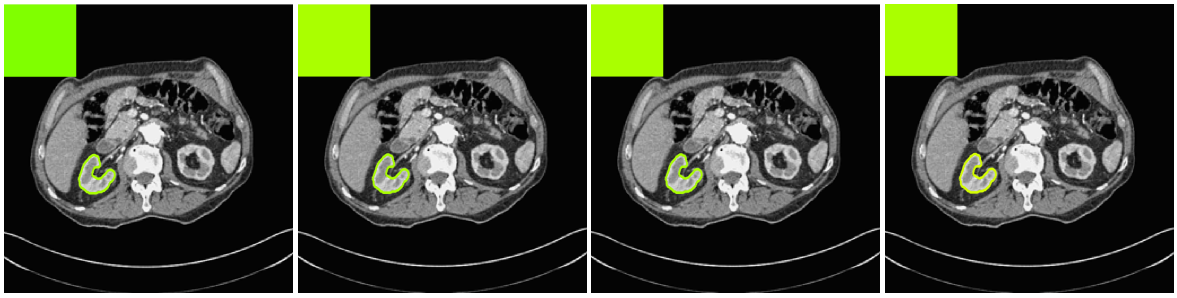
Table 4.1: Optimal TC values for Test Images 1–9, from the heatmaps in Figs. 4.11, 4.13, and 4.15. The best result for each image is given in bold.

to computational constraints, we run for each integer $\tilde{\lambda}, \theta$ between 1 and 10, and every fifth from 15 to 50. This aspect of a model's performance is vital when used in practice. The less sensitive to parameter choices a model is the more relevant it is in relation to potential applications.

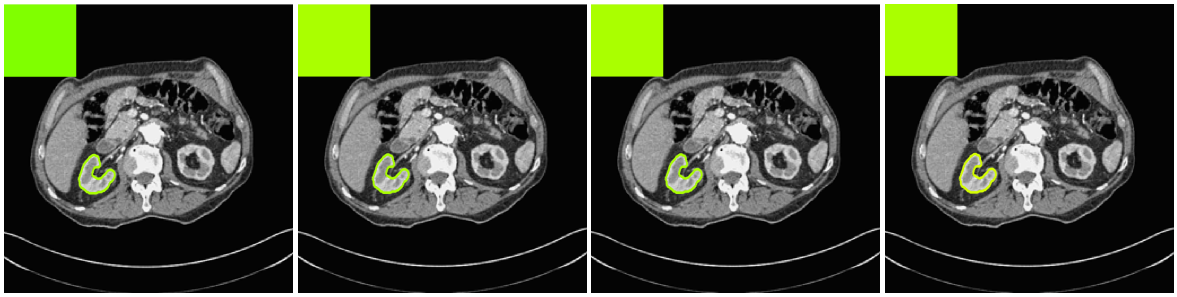
The TC values for the parameter sets $(\tilde{\lambda}, \theta)$ are presented as heatmaps in Figs. 4.11–4.15. A heatmap is a convenient way to display accuracy results for hundreds of tests concisely. In Fig. 4.9 we give an example heatmap with the same axes used for those in Figs. 4.11–4.15. For each of the combinations of parameter values $(\tilde{\lambda}, \theta)$ we give the TC value of the segmentation result and represent it by the appropriate colour. The



(a) $\tilde{\lambda}$ = 1, TC = 0.00. (b) $\tilde{\lambda}$ = 2, TC = 0.79. (c) $\tilde{\lambda}$ = 3, TC = 0.91. (d) $\tilde{\lambda}$ = 4, TC = 0.95.



(e) $\tilde{\lambda}$ = 5, TC = 0.95. (f) $\tilde{\lambda}$ = 6, TC = 0.95. (g) $\tilde{\lambda}$ = 7, TC = 0.94. (h) $\tilde{\lambda}$ = 8, TC = 0.94.



(i) $\tilde{\lambda}$ = 9, TC = 0.93. (j) $\tilde{\lambda}$ = 10, TC = 0.93. (k) $\tilde{\lambda}$ = 15, TC = 0.93. (l) $\tilde{\lambda}$ = 20, TC = 0.85.

Figure 4.10: Segmentation results and TC values for the proposed model whilst varying $\tilde{\lambda}$ (with $\theta = 4$). The colours correspond to the TC value (green is TC = 1, red is TC = 0), consistent with the scale in Fig. 4.8. This is for Test Image 5, with the corresponding heatmap provided in Fig. 4.13.

corresponding colour scale is shown in Fig. 4.8. Qualitatively, the more green areas of the heatmap the more accurate the model is for a wider set of parameters. Example results for Test Image 5 when varying $\tilde{\lambda}$ (with $\theta = 4$) for the proposed model are given

in Fig. 4.10. Here it can be seen what each accuracy result corresponds to visually.

Note. The axes have been removed from the heatmaps in Figs. 4.11–4.15 for presentational clarity. However, to be explicit, the axes used in all heatmaps are the same as those in Fig. 4.9.

Synthetic Images. These results are presented in Fig. 4.11. For Test Images 1–2 we see poor parameter robustness from all competing models, except for GAV which performs reasonably well. However, the proposed model has minimal parameter sensitivity for these images, with good results achieved for almost every combination of values tested. For Test Image 3 all models have a reasonable parameter range (except for RSF), however, the proposed model gives better quality results for a wider parameter range. The other models achieve reasonable results here as the foreground intensity of the ground truth is greater than the background ($c_1 = 0.75, c_2 = 0.49$), whereas for Test Images 1–2 they are equal ($c_1 = c_2 = 0.50$). These results highlight the key advantage of the proposed model.

Real Images. In Fig 4.13 we present results for Test Images 4–6. Here, the proposed model performs in a similar way to its competitors because these images are more typical selective segmentation problems in the sense that there is a clear distinction between the foreground and background intensities. In particular, the values in each case are: Test Image 4 ($c_1 = 0.85, c_2 = 0.25$), Test Image 5 ($c_1 = 0.70, c_2 = 0.19$), and Test Image 6 ($c_1 = 0.73, c_2 = 0.20$). It can be seen that the proposed model is competitive compared to previous approaches. The performance is quite poor for Test Image 5, but is arguably still the best for this challenging case. In Fig. 4.15 we present results for Test Images 7–9. Here the proposed model outperforms previous approaches significantly for each image. This is mainly due to the type of image considered. Specifically, the true intensities are: Test Image 7 ($c_1 = 0.12, c_2 = 0.24$), Test Image 8 ($c_1 = 0.10, c_2 = 0.23$), and Test Image 9 ($c_1 = 0.08, c_2 = 0.14$). The proposed model is capable of achieving results where $c_1 \approx c_2$, with other models failing completely in these cases.

4.7.2. Accuracy Comparisons

Here we aim to address the question of whether each model is capable of achieving an accurate result. In other words, assuming that factors such as parameter and user

input sensitivity are ignored, how successful is each approach. In Table 4.1 we present the optimal TC values for each model found from the tests described in the previous section, with the highest value in bold. Immediately we can see that the proposed model outperforms all other models in terms of accuracy for all test images (RSF equals it for Test Image 1). Below we will discuss some relevant details of the results, again by splitting the test images into synthetic and real.

Synthetic Images. We observe that for Test Images 1 and 2 (where $c_1 = c_2$, CV, LCV, and HYB fail completely. RSF and GAV perform well, with the proposed model being the most accurate with perfect results. For Test Image 3, all models are capable of achieving a good result. It should be noted that in this case $c_1 = 0.75$ and $c_2 = 0.49$. This difference enables the other models to perform well, although the proposed model is slightly superior with a perfect result.

Real Images. In Table 4.1 we can see that the proposed model is the most successful in terms of optimal accuracy. It is worth noting some inconsistency in the other models, with all but GAV having results that fall below TC $= 0.9$ for at least one image. GAV performs well for Test Images 4–9, with the proposed model slightly outperforming it in each case. It is worth reminding the reader that for GAV the parameters (β_1, β_2) have been refined for each example. Fixing this results in more variability in the quality of results. The proposed model has no such parameter optimisation between examples. We present the optimal results for Test Image 9 in Fig. 4.17. Here we can see how much variation there is in the quality of results for this lung CT image. GAV is the most competitive (TC $= 0.919$), but is visually inadequate. Two other models (CV, HYB) fail completely. In this case, the problem looks quite straightforward and yet other fitting terms are insufficient to produce a good result. Again, the proposed model tends to be superior in cases where $c_1 \approx c_2$ and is capable of achieving very good results for all the images considered. This highlight the advantages of the proposed fitting term.

4.7.3. User Input Randomisation

One key consideration for the practical use of selective segmentation models is that the result is not too reliant on user input. With intricate user input, accurate results are almost guaranteed. However, the benefit of this kind of approach is that accuracy should be attainable with minimal, intuitive user input. One challenge in this setting is how to

ascertain to what extent a method is dependent on the user input. In this section, we will generalise the user input for the proposed model in order to determine how sensitive it is in this respect. By generalising in this way we will make two assumptions about the markers, $\mathcal{M}$, consistent with the above considerations:

(i) All points are within the target object.

(ii) Only 3 markers are selected.

We regard neither of these assumptions to be too onerous on a user and are quite consistent with practical use. To perform this test, we randomly choose 1000 sets of 3 marker points and run each algorithm using them. The parameters $\tilde{\lambda}$ and θ are fixed at those which gave the optimal TC values in Table 4.1. For each set of marker points, we compute the corresponding TC value of applying the proposed model with this input. The results for each image are summarised by boxplots in Fig. 4.18 with examples of the worst results, excluding outliers, shown in Fig. 4.19. Here, it can be seen that the worst result outperforms the optimal results of the alternative models considered, which is impressive. Below we discuss the results for the test images, by again splitting them into synthetic and real images.

Synthetic Images. For the Test Images 1–3 we achieve near perfect segmentations in all cases, shown by the mean TC being between 0.99 and 1.00 in all cases (for Test Image 1, the mean is precisely 1.00) and a small variance around the mean. Therefore, we can conclude that for images of this type, where the foreground is homogeneous, our method is very robust to user input. Essentially, any reasonable set of markers should produce excellent results. It should be noted that the optimal results from comparable approaches are less than the mean result of 1000 random tests for our method. This can be observed in Table 4.1. Furthermore, these methods often fail completely. This is a key result highlighting the advantages of our method. In visually simple cases (Test Images 1–3) our new data fitting term is an improvement on existing approaches by modifying the underlying assumptions involved.

Real Images. In all cases for Test Images 4–9 the mean values show that the segmentation results are highly accurate. Also, we notice that the variances are very reasonable demonstrating the robustness of varying the user input. This is an important aspect of selective segmentation and highlights the advantages of the proposed fitting term. For Test Images 4–6 we observe more variability in the accuracy due to minor intensity in-

homogeneity in the foreground. This means randomising the user input will be more sensitive. However, we can see that the results are very good with the mean accuracy being competitive with the optimal accuracy of comparable methods. In the case of the lung CT images (Test Images 7–9), the variance in TC values is very small due to the homogeneity of the foreground. Again, it is important to compare the results of 1000 random results using our proposed model to the optimal result of comparable methods. For these images, all of the methods except GAV have at least one TC value below 0.9. However, GAV requires the tuning of additional parameters (β_1, β_2) whilst the proposed model does not. Compared to GAV, we can see that the mean of our tests is similar to the optimal value of GAV. One exception is for Test Image 9 (shown in Fig. 4.17), where there is a significant gap in favour of our model. Again, from Fig. 4.19, we can see that the worst result of randomising the user input for the proposed model is competitive with the optimal results of the alternatives. This is one of the most encouraging aspects of the tests; the proposed model is remarkably robust to varying user input. This proves that successful results with minimal, intuitive user input is possible for a range of examples.

4.7.4. Additional Tests

In order to further establish the robustness of our method, we now introduce the results of testing our approach against competing interactive segmentation methods. Specifically, we compare against the work of Nguyen et al. [121] and Dong et al. [61], referred to as CAC and SRW respectively and detailed in §4.3. The results are presented in Fig. 4.20, showing a boxplot of accuracy in terms of TC on a set of 30 CT images (excluding outliers). The target structure we consider is the spleen, as this consists of a relatively homogeneous foreground, appropriate for the approach considered. The data has been manually contoured providing ground truth data for the image set. We compare CAC [121] and SRW [61] against our method with five variations of user input for each image. A representative example for three images is shown in Fig. 4.21. This shows foreground (red) and background (blue) user input regions. For our method, we define the red region as $\mathcal{P}$ as discussed in §4.1 and enforce hard constraints on the blue region. We refer to the results of the proposed approach using this input as Ours (i). We also include results of randomising the user input in an identical way to §4.7.3. For each image, we generate 1000 simulated user input choices, which we present as Ours (ii). It is impor-

tant to note that the difference between Ours (i) and (ii) is only the definition of $\mathcal{P}$. The method and parameters are fixed between each.

The performance of CAC [121] is very good, as shown in Fig. 4.20. We have included an additional figure to highlight the difference between CAC and Ours (i) and (ii) more precisely. This is shown in Fig. 4.22 (this is the same as Fig. 4.20 with TC restricted to [0.8,1]). Here we can see that the proposed approach has a slightly better median (0.96 compared to 0.94) and is generally more consistent than CAC. This is particularly evident when considering the worst TC results of CAC (0.19) against ours (0.87).

In Fig. 4.20 it can be seen that our method exceeds the performance of SRW by a large margin (0.66 compared to 0.95). One possible reason for this is that the input used, as displayed in Fig. 4.21, is restricted to be as intuitive as possible. SRW is capable of achieving improved results with more elaborate foreground/background input. However, it is generally reliant on a trial and error approach which is not ideal in practice. This highlights an important advantage of our method. It is able to achieve a high standard of results with simple user input. This is reinforced by considering Ours (ii), where the results of 30000 random variations of the user input does not cause a drop off in accuracy compared to the 150 manual user input selections. Again, this can be seen more clearly in Fig. 4.22. In fact, the results for the proposed approach with the random input are slightly better than with the manual input. This underlines the robustness to user input in the model, which is a vital aspect of selective segmentation.

4.8. Conclusion

In this chapter, we have proposed a new intensity fitting term, for use in selective segmentation. We have compared it to fitting terms from comparable approaches (CV, RSF, LCV, HYB, GAV), in order to address an underlying problem in selective segmentation: if the foreground is approximately homogeneous what is the best way to define the intensity fitting term? Previous methods [135, 137, 153] involve contradictions in the formulation, which we attempt to address.

We have evaluated the success of the proposed model in four respects: parameter robustness, optimal accuracy, dependence on user input, and comparisons to competing selective models. Our focus is on medical applications, where the target object has ap-

proximately homogeneous intensity. In each way, the proposed model performs very well, particularly in cases where the true foreground and background intensities are similar. We have shown that our method is remarkably insensitive to varying user input, highlighting its potential for use in practice, and also outperforms competitive algorithms in the literature.

Figure 4.11: Heatmaps of TC values for permutations of $\tilde{\lambda}$ and θ. Each row and column is labelled according to the model used and the image tested. The colour is consistent with the scale in Fig. 4.8. Here, we present Test Images 1 – 3.

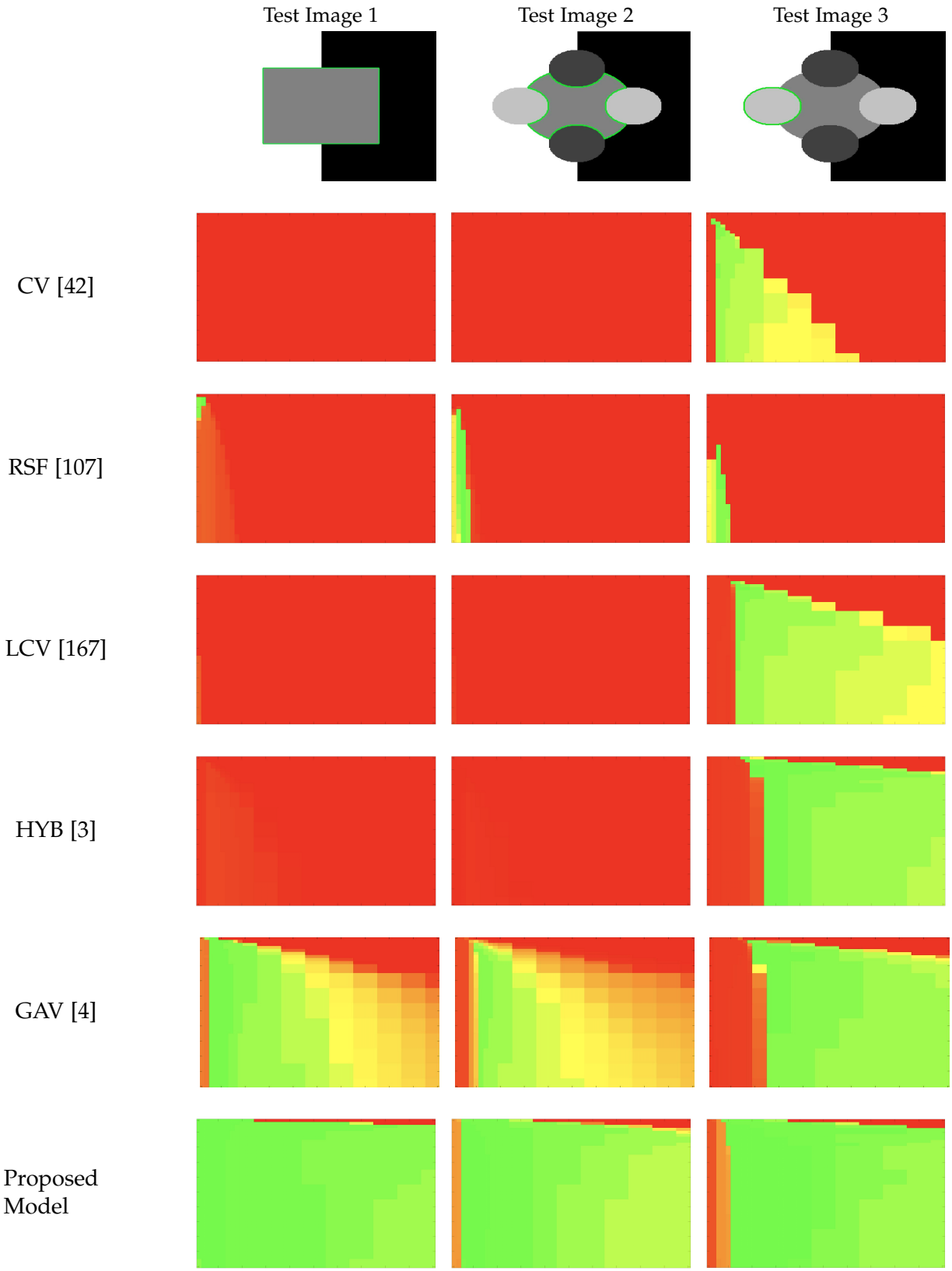


Figure 4.13: Heatmaps of TC values for permutations of $\tilde{\lambda}$ and θ. Each row and column is labelled according to the model used and the image tested. The colour is consistent with the scale in Fig. 4.8. Here, we present Test Images 4 – 6.

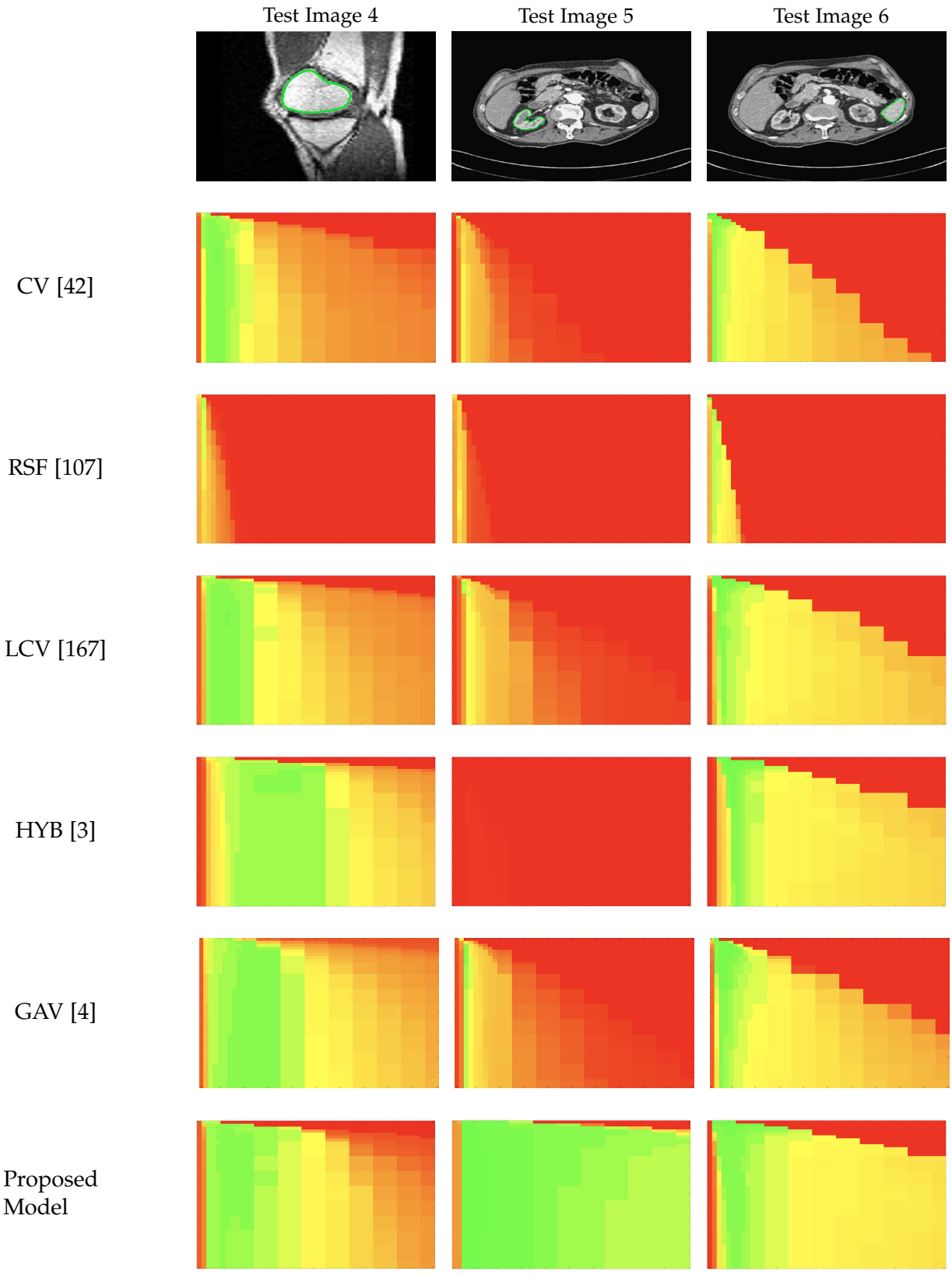
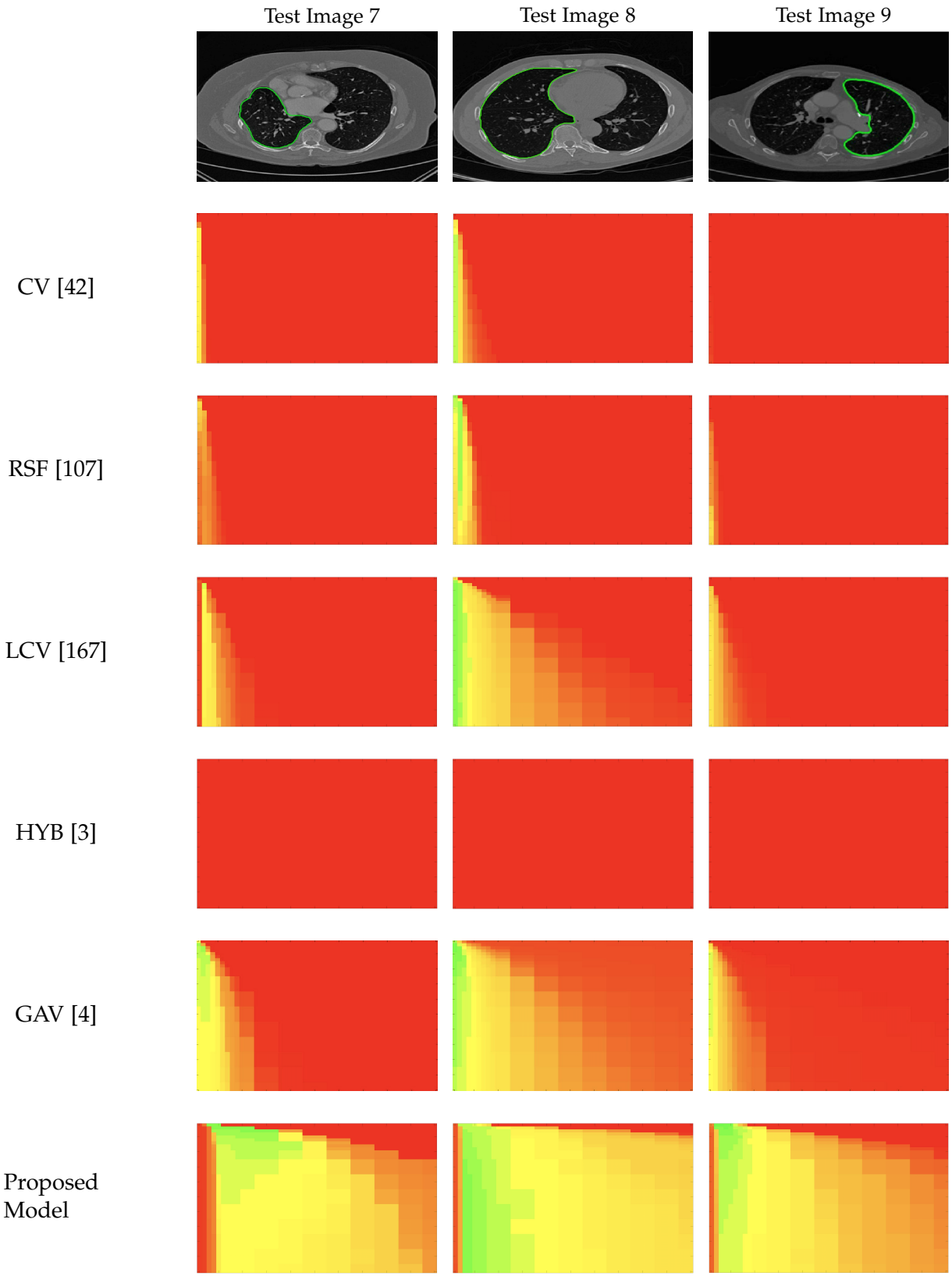
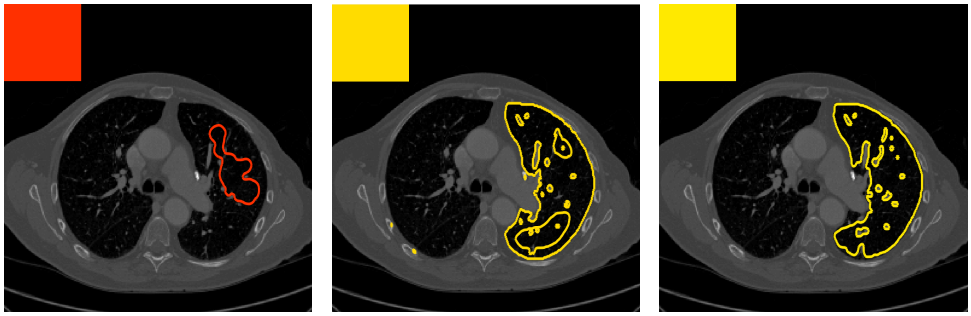


Figure 4.15: Heatmaps of TC values for permutations of $\tilde{\lambda}$ and θ. Each row and column is labelled according to the model used and the image tested. The colour is consistent with the scale in Fig. 4.8. Here, we present Test Images 7 – 9.





(a) CV [42], TC = 0.18. (b) RSF [107], TC = 0.78. (c) LCV [167], TC = 0.83

(d) HYB [3], TC = 0.00. (e) GAV [4], TC = 0.92. (f) Proposed, TC = 0.97

Figure 4.17: We present the optimal result for each model for Test Image 9. The accuracy is represented by colour, consistent with the scale in Fig. 4.8. The proposed model significantly outperforms previous approaches in this case.

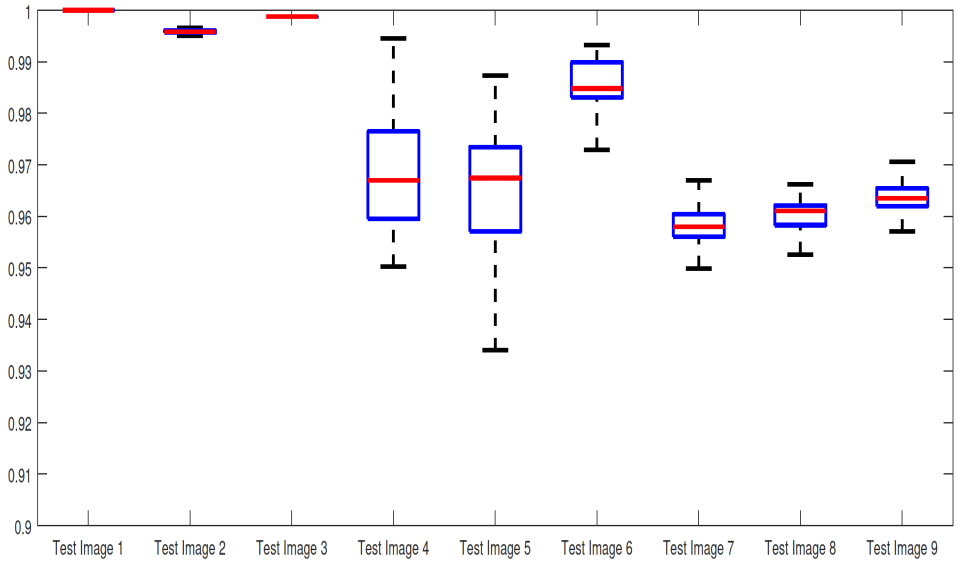
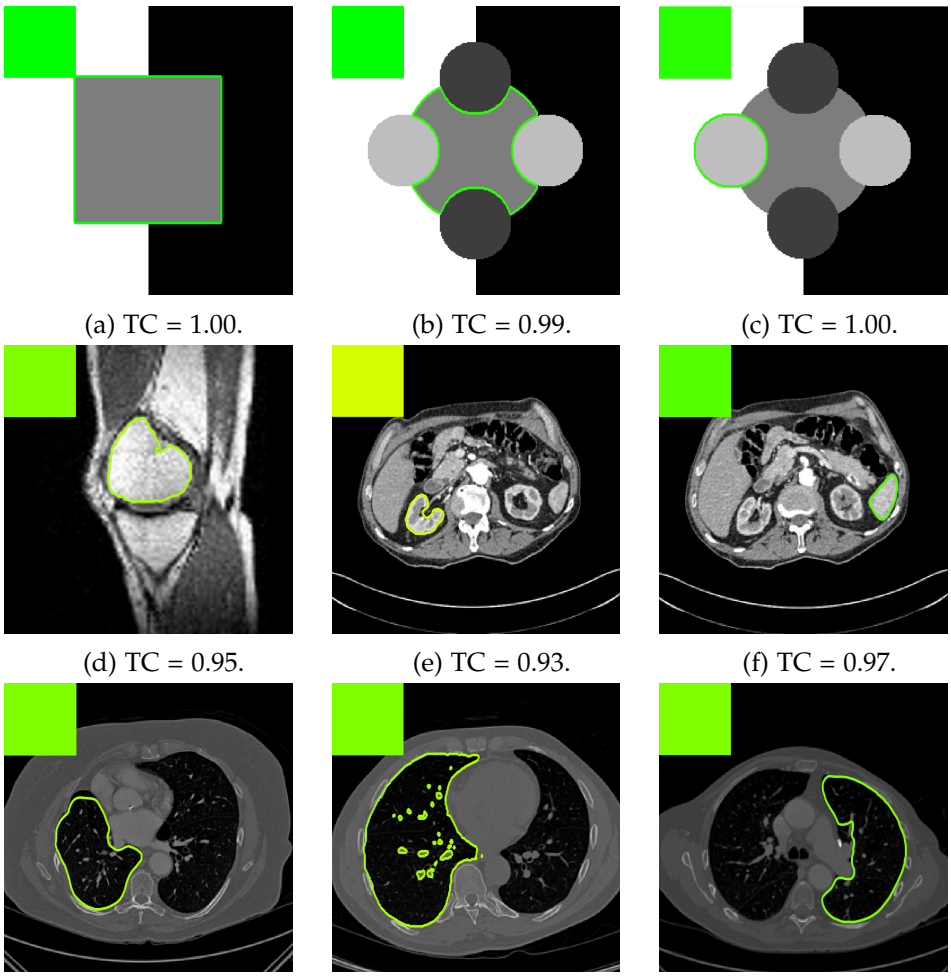


Figure 4.18: Boxplots of the TC values for 1000 random user inputs using the proposed model. We observe that the method is remarkably consistent. Even the worst results, excluding outliers, are competitive with the optimal results of the existing approaches shown in Table 4.1. Note that the whiskers extend to the largest and smallest values excluding outliers.



(a) TC = 1.00. (b) TC = 0.99. (c) TC = 1.00.

(d) TC = 0.95. (e) TC = 0.93. (f) TC = 0.97.

(g) TC = 0.95. (h) TC = 0.95. (i) TC = 0.96.

Figure 4.19: Results for the proposed model for each image, including TC values. The worst result, excluding outliers, of 1000 random user inputs for each example is presented. This demonstrates that the model is robust to user input, with poor results being competitive with the optimal result of competitors.

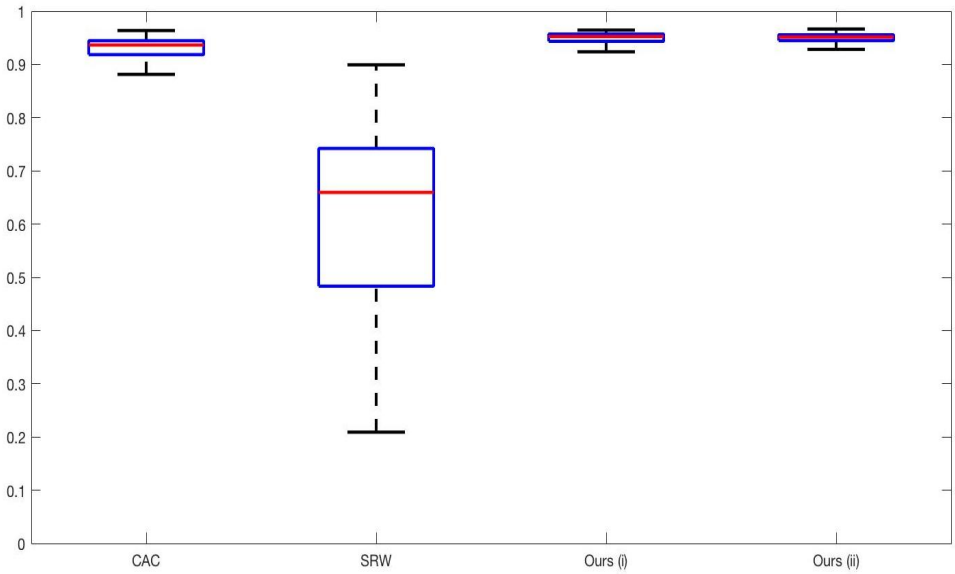


Figure 4.20: Boxplots of the TC values comparing our method to CAC [121] and SRW [61] for 30 test images. Ours (i) refers to using identical user input to CAC and SRW, with a sample shown in Fig. 4.21. Ours (ii) refers to 1000 random variations of the user input for each image.

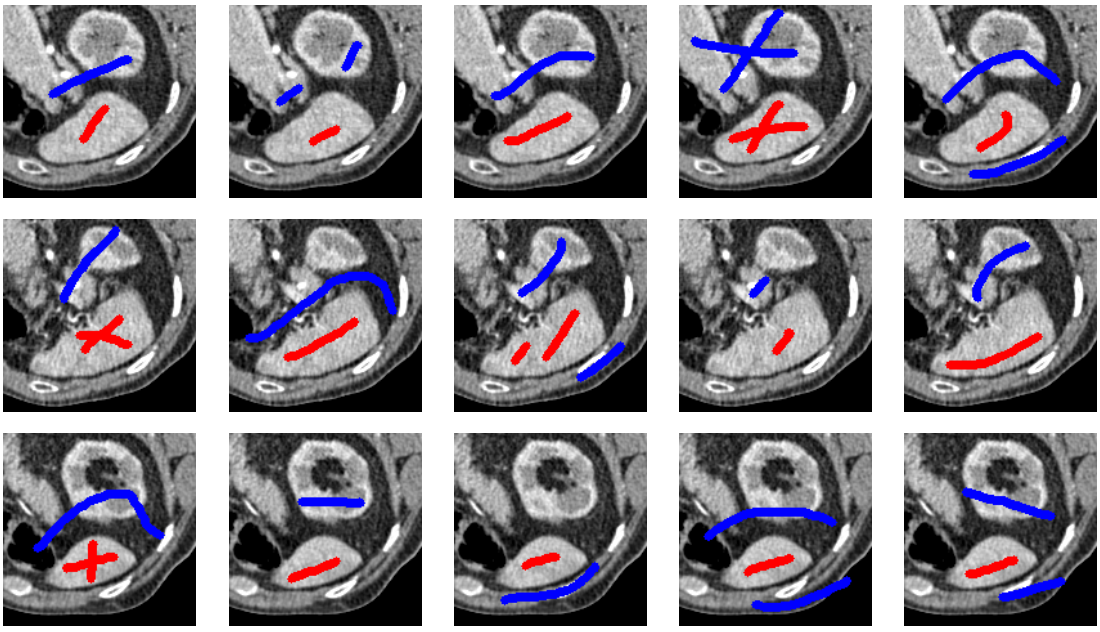


Figure 4.21: Examples of the input used to compare our method to CAC [121] and SRW [61]. Each row represents an image in the dataset and we present five variations of the input used in the tests described in §4.7.4.

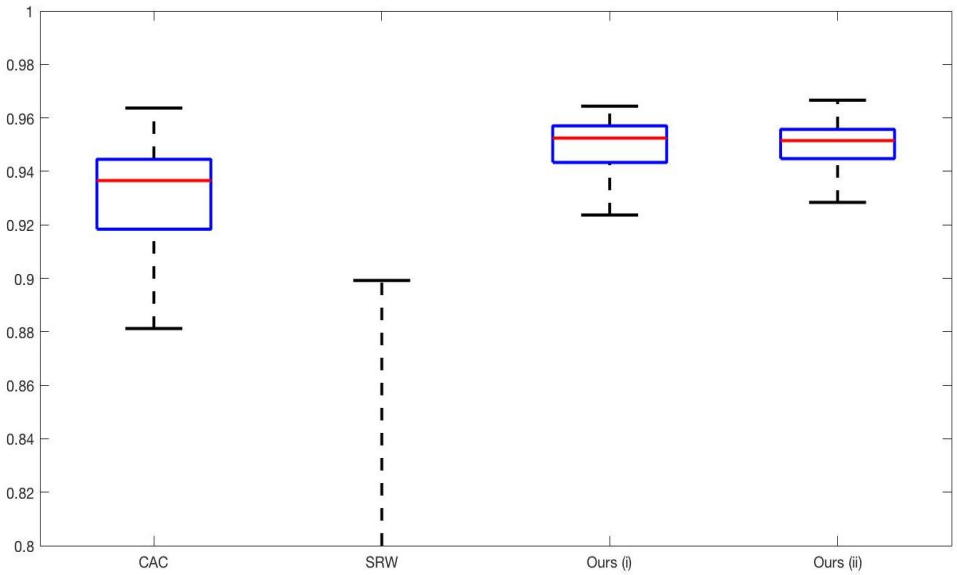


Figure 4.22: Boxplots of the TC values from Fig. 4.20 for TC $\in [0.8, 1]$. Here, the extent to which the proposed method outperforms CAC [121] is clearer for both types of input.

Chapter 5

Multigrid Algorithm Based on Hybrid Smoothers for Variational and Selective Segmentation Models

This chapter introduces two non-standard smoothers which outperform the standard smoothers used in non-linear multigrid schemes for PDEs with highly variable or discontinuous coefficients. These new schemes even outperform the line smoothers. This chapter is based on the paper written by the author [138].

5.1. Introduction

Segmentation of an image into its individual objects is one incredibly important application of image processing techniques. Not only are accurate segmentation results required, but it is also required that the segmentation method is fast. Many imaging applications demand increasingly higher resolution e.g. an image of size 25000×25000 (or practically 10^8 unknowns) can be common in oncology imaging. Here we address the problem of slow solutions by developing a fast multigrid method for PDEs arising from segmentation models.

Solving the PDE models quickly, in the context of large-scale images, remains a chal-

lenge. The variational approach to image segmentation involves the minimisation of an energy functional such as that in [135]. This will typically involve solving a system of equations from a discretised PDE using an iterative method. In particular, discretisations of models such as [12, 13, 46, 135, 153] are non-linear and so require non-linear iterative methods to solve. The number of equations in the system is equal to the number of pixels in the image, which can be very large, and for each equation in the system, the number of steps of an iterative method required can also be very large (to reach convergence). Due to improvements in technology and imaging, we now can produce larger and larger images, however, this has the direct consequence that analysis of such images has become much more computationally intensive. We remark that if we directly discretise the variational models first (without using PDEs), Chan-Vese type models can be reformulated into minimisation based on graph cuts and then fast algorithms have been proposed [14, 112].

The multigrid approach for solving PDEs in imaging has been tried before and previous work by Badshah and Chen [10, 11] introduced a 2D Chan-Vese multigrid algorithm for two-phase and multi-phase images. Additionally, Zhang et al. [176] implemented a multigrid algorithm for the 3D Chan-Vese model. The fundamental idea behind multigrid is that if we perform most of the computations on a reduced resolution image, then the computational expense is lower. We then transfer our solution from the low-resolution grid to the high-resolution grid through interpolation and smooth out any errors which have been introduced by the interpolation using a few steps of a smoothing algorithm, e.g. Gauss-Seidel. The multigrid method is an optimal solver when it converges [109, 158]. This requires that the smoothing scheme, which corrects the errors when transferring between the higher and lower resolution images and vice-versa, is effective, i.e. reduces the error magnitude of high-frequency components quickly.

In the large literature of multigrid methods, the convergence problem associated with non-smooth or jumping coefficients was often highlighted [2, 32] and developing working algorithms which converge is a key problem. Much attention was given to designing better coarsening strategies and improved interpolation operators [164, 181] while keeping the simple smoothers; such as the damped Jacobi, Gauss-Seidel or line smoothers. In practice, one can quickly exhaust the list of standard smoothers and yet cannot find a suitable one unless compromising in optimality by increasing the number of iterations. In contrast, our approach here is to seek a non-standard and more effective smoother

with an acceptable smoothing rate. Our work is motivated by Napov and Notay [120] who established the explicit relationship of a smoothing rate to the underlying multigrid convergence rate for linear models; in particular the former also serves as the lower bound for the latter.

The contributions of this chapter can be summarised as follows:

1. We review six smoothers for the Rada-Chen and Spencer-Chen selective segmentation models and perform Local Fourier Analysis (LFA) to assess their performance and quantitatively determine their effectiveness (or lack of).
2. We propose an effective non-linear multigrid method to solve the Rada-Chen model [135] and the Spencer-Chen model [153], based on new smoothers that add non-standard smoothing steps locally at coefficient jumps. We recommend in particular one of our new hybrid smoothers which achieves a better smoothing rate than the other smoothers studied and thus gives rise to a multigrid framework which converges to the energy minimiser faster than when standard smoothers are used.

It is worth noting that this new multigrid algorithm doesn't converge for models such as the convex Spencer-Chen model [153] or the convex models introduced in Chapters 3 and 4. This is due to instabilities in the convex penalty term $\nu'(u)$ in the functionals of those models. For a small change in u, we have a large change in the value of the penalty term. Even with regularisation the smoother is unstable and doesn't converge. Therefore, the overall multigrid algorithm doesn't converge. The authors address this problem in [139] and give a convergent multigrid algorithm for models including the convex relaxation term $\nu(u)$. Therefore, to be clear, the techniques of this chapter are highly applicable to standard non-convex segmentation models, we exemplify this by considering the Rada-Chen and Spencer-Chen models.

The remainder of this chapter is structured as follows; in §5.2, give the finite difference discretisation that we use for the Rada-Chen and Spencer-Chen models introduced earlier in §2.2.7. In §5.3, we describe the Full Approximation Scheme multigrid framework, give details of six smoothers that we consider and compare the smoothing rates. We find that none of these standard smoothers can produce a small enough smoothing rate to yield an effective multigrid method and so in §5.4, we then introduce two new hybrid smoothers based on new iterative schemes to improve the smoothing rates at those pixels where the six smoothers perform badly. In §5.5, we test our algorithms with some

numerical results, recommend the best algorithm using one of our proposed smoothers and analyse the complexity of the recommended multigrid algorithm. Finally, in §5.6, we provide some concluding remarks.

5.2. The Rada-Chen and Spencer-Chen Models

In this section, we will repeat the Rada-Chen and Spencer-Chen models introduced earlier in §2.2.7 and give the discretisation of the models obtained using finite differences. These discretised versions of the PDEs will be solved by the non-linear multigrid algorithm discussed later.

The Rada-Chen model [135]. This is the first model we focus on in this chapter, defined by

$$\begin{aligned} F_{RC}(\phi, c_1, c_2) =& \mu \int_\Omega d(x,y) g(|\nabla z(x,y)|^2) |\nabla H_\varepsilon(\phi)| dxdy \\ &+ \lambda_1 \int_\Omega (z(x,y) - c_1)^2 H_\varepsilon(\phi) dxdy + \lambda_2 \int_\Omega (z(x,y) - c_2)^2 (1 - H_\varepsilon(\phi)) dxdy \\ &+ \nu \left[\left(\int_\Omega H_\varepsilon(\phi) dxdy - A_1 \right)^2 + \left(\int_\Omega (1 - H_\varepsilon(\phi)) dxdy - A_2 \right)^2 \right], \end{aligned} \tag{5.1}$$

where $\mu, \lambda_1, \lambda_2, \nu$ are fixed non-negative parameters. The distance term, given by $d(x,y)$ imposes a penalty for the contour deviating from the marker set – various options for $d(x,y)$ are given in §2.2.7.2. In the Rada-Chen paper the authors use

$$d(x,y) = \prod_{i=1}^{k} \left(1 - \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{x}_i|^2}{2\sigma^2} \right) \right)$$

where σ is a non-negative tuning parameter. The value of σ has a large impact on the resulting distance map. It can be set manually and tuned to the image being segmented, or automatically, e.g. $\sigma = \min_{i,j,i \neq j} |\boldsymbol{x}_i - \boldsymbol{x}_j|$. The edge detector function is given by

$$g(|\nabla z(x,y)|) = \frac{1}{1 + \beta |\nabla z(x,y)|^2}$$

for tuning parameter β which takes value 0 at edges and is 1 away from them. A_1 is the area of the polygon formed from the points of $\mathcal{S}$ and $A_2 = |\Omega| - A_1$. The final term of this functional therefore puts a penalty on the area inside a contour being very different to A_1. The first variation of (5.1) with respect to ϕ gives the Euler-Lagrange form [135]

$$\begin{aligned}\delta_\varepsilon(\phi)\Bigg\{\mu\nabla\cdot\left(\frac{d(x,y)\cdot g(|\nabla z(x,y)|^2)\nabla\phi}{|\nabla\phi|}\right) - \left[\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2\right] \\ - \nu\left[\left(\int_\Omega H_\varepsilon(\phi)dxdy - A_1\right) - \left(\int_\Omega(1-H_\varepsilon(\phi)) - A_2\right)\right]\Bigg\} = 0,\end{aligned} \tag{5.2}$$

in Ω with the condition that $\frac{\partial\phi}{\partial\boldsymbol{n}} = 0$ on $\partial\Omega$, $\boldsymbol{n}$ the outward normal vector and $\delta_\varepsilon(\phi) = \frac{dH_\varepsilon(\phi)}{d\phi}$.

Discretisation of the Rada-Chen model. We denote by $\phi_{i,j} = \phi(x_i, y_j)$ the approximation of ϕ at (i,j) for $1 \le i \le n$ and $1 \le j \le m$. We let h_x and h_y be the grid spacings in the x and y directions respectively. Using finite differences, and noting $A_2 = 1 - A_1$, we obtain the scheme

$$\begin{aligned}&A_{i,j}\phi_{i+1,j} + B_{i,j}\phi_{i-1,j} + C_{i,j}\phi_{i,j+1} + D_{i,j}\phi_{i,j-1} - S_{i,j}\phi_{i,j} \\ &-\delta_\varepsilon(\phi_{i,j})\left\{\left[\lambda_1(z_{i,j}-c_1)^2 - \lambda_2(z_{i,j}-c_2)^2\right] - 2\nu\left[h_xh_y\sum_{k,l}H_\varepsilon(\phi_{k,l}) - A_1\right]\right\} = 0,\end{aligned} \tag{5.3}$$

where $$G_{i,j} = \frac{d_{i,j}\cdot g(|\nabla z_{i,j}|)}{|\nabla\phi_{i,j}|}, \quad A_{i,j} = \frac{\mu\delta_\varepsilon(\phi_{i,j})}{h_x^2}G_{i+\frac{1}{2},j}, \quad B_{i,j} = \frac{\mu\delta_\varepsilon(\phi_{i,j})}{h_x^2}G_{i-\frac{1}{2},j},$$

$$C_{i,j} = \frac{\mu\delta_\varepsilon(\phi_{i,j})}{h_y^2}G_{i,j+\frac{1}{2}}, \quad D_{i,j} = \frac{\mu\delta_\varepsilon(\phi_{i,j})}{h_y^2}G_{i,j-\frac{1}{2}}, \quad S_{i,j} = A_{i,j} + B_{i,j} + C_{i,j} + D_{i,j}, \tag{5.4}$$

This equation is highly non-linear as all coefficient terms $A_{i,j}, B_{i,j}, C_{i,j}$ and $D_{i,j}$ depend on ϕ both in the G component but also within the δ component.

The Spencer-Chen model [153]. The second model we focus on in this chapter is defined by

$$\begin{aligned}F_{SC}(\phi,c_1,c_2) =& \mu\int_\Omega g(|\nabla z(x,y)|^2)|\nabla H_\varepsilon(\phi)|dxdy + \lambda_1\int_\Omega(z(x,y)-c_1)^2H_\varepsilon(\phi)dxdy \\ &+ \lambda_2\int_\Omega(z(x,y)-c_2)^2(1-H_\varepsilon(\phi))dxdy + \theta\int_\Omega d(x,y)H_\varepsilon(\phi)dxdy,\end{aligned} \tag{5.5}$$

where $\mu, \lambda_1, \lambda_2$ and θ are fixed non-negative parameters. Note that this model differs from the Rada-Chen model (5.1) as the distance function has been separated from the edge detector term and is now a standalone penalty term. This model has Euler-Lagrange form

$$\delta_\varepsilon(\phi)\left\{\mu\nabla\cdot\left(\frac{g(|\nabla z(x,y)|^2)\nabla\phi}{|\nabla\phi|}\right)-\left[\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2\right]-\theta d(x,y)\right\}=0, \tag{5.6}$$

in Ω with the condition that $\frac{\partial\phi}{\partial\boldsymbol{n}}=0$ on $\partial\Omega$, again with $\boldsymbol{n}$ the outward normal vector. We discretise this similarly to the Rada-Chen model previously.

5.3. Non-linear multigrid Algorithm 1

Segmentation using a non-linear multigrid algorithm has been explored by Badshah and Chen [10, 11] for the Chan-Vese model [46] and the Vese-Chan model [162] which are global segmentation models. A multigrid method has not yet been applied to selective segmentation and this is the main task of this chapter, to apply the multigrid method to the Rada-Chen (5.1) and Spencer-Chen (5.5) selective segmentation models. However, as we will see shortly, the task is challenging as standard methods do not work. For brevity we will restrict consideration just to the Rada-Chen model as the derivations for the Spencer-Chen model are similar.

5.3.1. The Full Approximation Scheme

To solve the Rada-Chen model we must solve the non-linear system (5.3) and so we will use the non-linear Full Approximation Scheme [43, 49, 89, 158] algorithm due to Brandt [27]. Denote a discretised system by

$$N^h\phi^h=f^h, \tag{5.7}$$

where h indicates that these are the functions on the $n\times m$ cell-centred grid Ω^h and N^h is the discretised non-linear operator (which contains the boundary conditions). Similarly, define the grids Ω^{2h} as the $\frac{n}{2}\times\frac{m}{2}$ cell-centred grid resulting from the standard coarsening

[158] of Ω^h, we indicate functions on Ω^{2h} by f^{2h}, N^{2h} and ϕ^{2h}. Let Φ^h be an approximation to ϕ^h such that the error $e^h = \phi^h - \Phi^h$ is smooth. Define the residual as $r^h = f^h - N^h\Phi^h$. Therefore using (5.7) we have the residual equation

$$N^h(\Phi^h + e^h) - N^h\Phi^h = r^h.$$

If the error e^h is *smooth* then this can be well approximated on Ω^{2h}; the assumption can be a big issue for non-linear problems. With an approximation of e^h on Ω^{2h} we can solve the residual equation on Ω^{2h}, which is significantly less computationally expensive than solving on Ω^h, and then transfer this error to Ω^h and use it to correct the approximation Φ^h. This method, using the two grids Ω^{2h} and Ω^h, is called a two-grid cycle and it can be nested such that we can consider solving on $\Omega^{4h}, \Omega^{8h}, \ldots$ and transferring the errors up through the levels to Ω^h and smoothing on each level. This is the multigrid method. We transfer from Ω^h to Ω^{2h} by restriction and from Ω^{2h} to Ω^h by interpolation.

Restriction. We use the full-weighting restriction operator $I_h^{2h}\Phi^h = \Phi^{2h}$ given in §2.3.2.1.

Interpolation. We use the bilinear interpolation operator $I_{2h}^h\Phi^{2h} = \Phi^h$ given in §2.3.2.2.

We now move to the most important element of the multigrid method – the smoother. As previously mentioned, we need e^h to be smooth to ensure that Φ^h is a good approximation to ϕ^h. In practice, we smooth e^h by using an iterative method such as Gauss-Seidel [10, 11] and the success or failure of a multigrid method hinges on the effectiveness of it at smoothing the errors.

5.3.2. Smoothers for the Rada-Chen Model

Gauss-Seidel and Newton iterative methods have been shown to be effective smoothers for PDE problems with smooth coefficients [158, 164]. In this subsection, we look at three distinct smoothing iterative techniques; lexicographic Gauss-Seidel, line Gauss-Seidel and Newton smoothers. For each of these smoothers, we consider two different approaches for fixing the coefficients in the scheme - globally or locally. Hence overall we consider six smoothers for [135]; the same smoothers are adaptable for [153] in a simple way.

Smoothers 1-2 (GSLEX I - II). Lexicographic Gauss-Seidel smoothers are widely used in multigrid methods [10, 158]. We update $\phi_{i,j}$ one at a time and work across and down through the grid of pixels in an image.

Lexicographic Gauss-Seidel smoothers for the Rada-Chen model [135]. We can rearrange (5.3) as

$$\phi_{i,j} = \left(A_{i,j}\phi_{i+1,j} + B_{i,j}\phi_{i-1,j} + C_{i,j}\phi_{i,j+1} + D_{i,j}\phi_{i,j-1} - f_{i,j}\right) \,/\, S_{i,j}, \tag{5.8}$$

where

$$f_{i,j} = \delta_\varepsilon(\phi_{i,j})\Big\{\left[\lambda_1(z_{i,j}-c_1)^2 - \lambda_2(z_{i,j}-c_2)^2\right] + 2\nu\big[+h_x h_y \sum_{k,l} H_\varepsilon(\phi_{k,l}) - A_1\big]\Big\},$$

to obtain a fixed point scheme for the Rada-Chen model. There are two approaches for implementing this smoother; either update the coefficients globally at the start of each outer iteration or update them locally, immediately after solving for each pixel value. We denote the global smoother by GSLEX-I and the local smoother by GSLEX-II. In the algorithm for both smoothers, we cycle through each pixel (i,j) in turn solving (5.8) and updating the value of $\phi(i,j)$, only with GSLEX-II do we update the coefficients immediately and they are used in the update of $\phi(i,j)$ on the next iteration.

Smoothers 3-4 (GSLINE I - II). Line smoothers are often used for harder problems (e.g. discontinuous coefficients). Here we perform the Gauss-Seidel updates one column at a time but the approach can be easily reformulated for a row by row update.

Gauss-Seidel line smoothers for the Rada-Chen model [135]. If we rearrange (5.3) to have all the $\phi_{\cdot,j}$ terms on the left-hand side we obtain

$$A_{i,j}\phi_{i+1,j} + B_{i,j}\phi_{i-1,j} - S_{i,j}\phi_{i,j} = F_{i,j} = -C_{i,j}\,\phi_{i,j+1} - D_{i,j}\phi_{i,j-1} + f_{i,j}, \tag{5.9}$$

where we can reformulate (5.9) as the following tridiagonal system

$$\begin{bmatrix} -S_{1,j} & A_{1,j} & 0 & \dots & 0 & 0 \\ B_{2,j} & -S_{2,j} & A_{2,j} & \ddots & 0 & 0 \\ 0 & B_{3,j} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & A_{n-2,j} & 0 \\ 0 & 0 & \ddots & B_{n-1,j} & -S_{n-1,j} & A_{n-1,j} \\ 0 & 0 & \dots & 0 & B_{n,j} & -S_{n,j} \end{bmatrix} \cdot \begin{bmatrix} \phi_{1,j} \\ \phi_{2,j} \\ \vdots \\ \vdots \\ \phi_{n-1,j} \\ \phi_{n,j} \end{bmatrix} = \begin{bmatrix} F_{1,j} \\ F_{2,j} \\ \vdots \\ \vdots \\ F_{n-1,j} \\ F_{n,j} \end{bmatrix}. \tag{5.10}$$

This system is diagonally dominant (by definition (5.4)) and if $C_{i,j} + D_{i,j} \neq 0$ then the system is strictly diagonally dominant. We can choose parameters for the edge detector and distance function which ensure this is always true. Therefore this will ensure that the Gauss-Seidel line smoother will converge to a solution [73]. As before, we obtain two smoothers; the global smoother GSLINE-I and the local smoother GSLINE-II.

Smoothers 5-6 (NEWT I - II). Our last set of smoothers rely on the Newton fixed point iteration schemes.

Newton smoothers for the Rada-Chen model [135]. We can rewrite (5.3) in a non-linear form for $\phi_{i,j}$

$$S_{i,j}\phi_{i,j}^{(k)} - P_{i,j} + Q_{i,j}(\phi_{i,j}^{(k)}) = 0.$$

where

$$P_{i,j} = A_{i,j}\phi_{i+1,j} + B_{i,j}\phi_{i-1,j} + C_{i,j}\phi_{i,j+1} + D_{i,j}\phi_{i,j-1} - \delta_\varepsilon(\phi_{i,j}) \left[\lambda_1(z_{i,j} - c_1)^2 - \lambda_2(z_{i,j} - c_2)^2\right]$$

and $Q_{i,j} = 2\nu\delta_\varepsilon(\phi_{i,j}) \left[h_x h_y \sum_{k,l} H_\varepsilon(\phi_{k,l}) - A_1\right]$. The Newton scheme to compute $\phi_{i,j}^{(k+1)}$ is

$$\phi_{i,j}^{(k+1)} = \phi_{i,j}^{(k)} - \left(S_{i,j}\phi_{i,j}^{(k)} - P_{i,j} + Q_{i,j}(\phi_{i,j}^{(k)})\right) \,/\, \left(S_{i,j} + Q'_{i,j}(\phi_{i,j}^{(k)})\right) \tag{5.11}$$

where $Q'_{i,j}(\phi_{i,j}^{(k)}) = 2\nu\delta_\varepsilon(\phi_{i,j})^2 h_x h_y + 2\nu\delta'_\varepsilon(\phi_{i,j}))\left[h_x h_y \sum_{k,l} H_\varepsilon(\phi_{k,l}) - A_1\right]$. We again have a global smoother, NEWT-I, and a local smoother, NEWT-II.

5.3.3. Algorithm 1

In §5.3.1 we briefly discussed the FAS across two grids, Ω^h (the fine grid) and Ω^{2h} (the coarse grid). The two-grid cycles can be nested so we can perform the majority of the computations on coarser grids than Ω^{2h}, such as $\Omega^{4h}, \Omega^{8h}, etc$ and recursive use of V-cycles gives rise to multigrid schemes [158]. The general non-linear multigrid Full Approximation Scheme algorithm is given by Algorithm 1.

Algorithm 1: Full Approximation Scheme, $\phi^h \leftarrow$ FASMG(ϕ^h, $\mathcal{N}^h$, f^h, η, ν_1, ν_2, level, max_level, Smoother)

Pre-smoothing: Perform ν_1 iterations of the smoother: $\tilde{\phi}^h \leftarrow Smoother(\phi^h, f^h, \nu_1)$.

Coarse grid correction: Compute the residual: $r^h = f^h - N^h\tilde{\phi}^h$.
Transfer the residual to Ω^{2h} by restriction: $r^{2h} = I_h^{2h} r^h$.
Compute: $\phi^{2h} = I_h^{2h}\tilde{\phi}^h, \Phi^{2h} = \phi^{2h}, f^{2h} = N^{2h}\phi^{2h} + r^{2h}$.
if $level = max\_level$ **then**
  Compute the exact solution ϕ^{2h} of $N^{2h}\phi^{2h} = f^{2h}$
  on Ω^{2h} using an accurate solver.
else
  Perform η cycles (steps) of
  $\phi^{2h} \leftarrow$ FASMG(ϕ^{2h}, N^{2h}, f^{2h}, η, ν_1, ν_2, level+1, max_level, Smoother).
end if

Interpolation: Compute: $e^{2h} = \phi^{2h} - \Phi^{2h}$.
Transfer the error to Ω^h by interpolation: $e^h = I_{2h}^h e^{2h}$.
Correct the fine grid approximation: $\hat{\phi}^h = \tilde{\phi}^h + e^h$.

Post-smoothing: Perform ν_2 iterations of the smoother: $\phi^h \leftarrow Smoother(\hat{\phi}^h, f^h, \nu_2)$.

5.3.4. Local Fourier Analysis of Algorithm 1 for the Rada-Chen Model

Local Fourier Analysis (LFA) is a useful tool for finding a quantitative measure for the effectiveness of a smoother [27, 49, 158]. It is designed to study linear problems with constant coefficients on an infinite grid. However, it is a standard and recommended [27, 32] tool to analyse non-linear operators. To overcome the limitations, we neglect the boundary conditions, extend the operator to an infinite grid and assume that we can linearise the operator locally (we do this by freezing the coefficients). LFA measures the

largest amplification factor on high-frequency errors, for example, if there is a smoothing rate of 0.8 this means that the high-frequency errors are damped by at least 20%. We initially must derive formulas for the approximation error at each pixel in our 5-point stencil.

Error forms. Using the definition of $f_{i,j}$, we can rewrite (5.3) as

$$A_{i,j}\phi_{i+1,j} + B_{i,j}\phi_{i-1,j} + C_{i,j}\phi_{i,j+1} + D_{i,j}\phi_{i,j-1} - S_{i,j}\phi_{i,j} = f_{i,j}, \tag{5.12}$$

where we fix $A_{i,j}, B_{i,j}, C_{i,j}$ and $D_{i,j}$ based on a previous iteration. The GSLEX I-II and NEWT I-II schemes all work in a lexicographic manner, and so if we denote the previous iteration as the k-th we can rewrite (5.12) as

$$A_{i,j}\phi^{(k)}_{i+1,j} + B_{i,j}\phi^{(k+1)}_{i-1,j} + C_{i,j}\phi^{(k)}_{i,j+1} + D_{i,j}\phi^{(k+1)}_{i,j-1} - S_{i,j}\phi^{(k+1)}_{i,j} = f_{i,j}, \tag{5.13}$$

and we obtain the error form by subtracting (5.13) from (5.12)

$$A_{i,j}e^{(k)}_{i+1,j} + B_{i,j}e^{(k+1)}_{i-1,j} + C_{i,j}e^{(k)}_{i,j+1} + D_{i,j}e^{(k+1)}_{i,j-1} - S_{i,j}e^{(k+1)}_{i,j} = 0, \tag{5.14}$$

Using a similar argument, we obtain the following error form for the line smoothers GSLINE I-II

$$A_{i,j}e^{(k+1)}_{i+1,j} + B_{i,j}e^{(k+1)}_{i-1,j} + C_{i,j}e^{(k)}_{i,j+1} + D_{i,j}e^{(k+1)}_{i,j-1} - S_{i,j}e^{(k+1)}_{i,j} = 0, \tag{5.15}$$

where $e^{(k)}_{i,j} = \phi_{i,j} - \phi^{(k)}_{i,j}$ and $e^{(k+1)}_{i,j} = \phi_{i,j} - \phi^{(k+1)}_{i,j}$.

Local Fourier Analysis. Define a general Fourier component by

$$F_{\theta_1,\theta_2}(x_i, y_j) = \exp\left(2\pi\mathbf{i}\frac{\theta_1 i}{n}\right) \cdot \exp\left(2\pi\mathbf{i}\frac{\theta_2 j}{m}\right) = \exp\left(\mathbf{i}\frac{\alpha_1 x_i}{h_x}\right) \cdot \exp\left(\mathbf{i}\frac{\alpha_2 y_j}{h_y}\right),$$

where $\alpha_1 = \frac{2\theta_1\pi}{n}$ and $\alpha_2 = \frac{2\theta_2\pi}{m}$ and $\mathbf{i}$ is the imaginary unit. Note that $\alpha_1, \alpha_2 \in [-\pi, \pi]$. If we assume for simplicity that the image is square and hence $n = m$, we first expand

$$e^{(k+1)}_{i,j} = \sum_{\theta_1,\theta_2=-n/2}^{n/2} \psi^{(k+1)}_{\theta_1,\theta_2} F_{\theta_1,\theta_2}(x_i, y_j), \qquad e^{(k)}_{i,j} = \sum_{\theta_1,\theta_2=-n/2}^{n/2} \psi^{(k)}_{\theta_1,\theta_2} F_{\theta_1,\theta_2}(x_i, y_j),$$

in Fourier components (see §2.3.1.3) and define the smoothing rate $\hat{\mu}_{i,j}$ by [158, 49]

$$\hat{\mu}_{i,j} = \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2) = \max_{\theta_1,\theta_2} \left| \frac{\psi^{(k+1)}_{\theta_1,\theta_2}}{\psi^{(k)}_{\theta_1,\theta_2}} \right|,$$

in the high-frequency range where $(\alpha_1,\alpha_2) = (\frac{2\theta_1\pi}{n}, \frac{2\theta_2\pi}{n}) \in [-\pi,\pi)^2 \backslash [-\frac{\pi}{2},\frac{\pi}{2})^2$. Since $\hat{\mu}_{i,j}$ is pixel dependent (non-linear problems), we may also call it the amplification factor associated with pixel (i,j).

Smoothing rates. For the GSLEX I-II, NEWT I-II smoothers, using (5.14) and (5.15), we obtain error amplification at pixel (i,j)

$$\hat{\mu}_{i,j} = \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2) = \max_{\alpha_1,\alpha_2} \left| \frac{A_{i,j}e^{\mathbf{i}\alpha_1} + C_{i,j}e^{\mathbf{i}\alpha_2}}{B_{i,j}e^{-\mathbf{i}\alpha_1} + D_{i,j}e^{-\mathbf{i}\alpha_2} - S_{i,j}} \right|,$$

and similarly for the GSLINE I-II smoothers we have

$$\hat{\mu}_{i,j} = \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2) = \max_{\alpha_1,\alpha_2} \left| \frac{C_{i,j}e^{\mathbf{i}\alpha_2}}{A_{i,j}e^{\mathbf{i}\alpha_1} + B_{i,j}e^{-\mathbf{i}\alpha_1} + D_{i,j}e^{-\mathbf{i}\alpha_2} - S_{i,j}} \right|. \tag{5.16}$$

Comparison of smoothing rates for all smoothers. We consider two different measures of the smoothing rates; the maximum and average over all pixels (i,j). We define these in the obvious way as

$$\tilde{\mu}_{\max} = \max_{i,j} \hat{\mu}_{i,j} = \max_{i,j} \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2) \quad \text{and} \quad \tilde{\mu}_{\text{avg}} = \frac{\sum_{i,j} \hat{\mu}_{i,j}}{n^2} = \frac{\sum_{i,j} \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2)}{n^2}.$$

Each of the smoothers was implemented in Algorithm 1 on the image in Figure 5.1(a) with a V-cycle ($\gamma = 1$) and using a 1024×1024 resolution image as the finest grid and a 32×32 image as the coarsest grid and in Table 5.1 we give $\tilde{\mu}_{\max}$ and $\tilde{\mu}_{\text{avg}}$ for the Rada-Chen and Spencer-Chen models.

In the spirit of previous works [10], for any of these smoothers, one would quote $\tilde{\mu}_{\text{avg}}$, and although this appears to be an excellent rate in all cases, it is the rate $\tilde{\mu}_{\max}$ that determines the multigrid convergence [120]. We, therefore, choose to focus on $\tilde{\mu}_{\max}$. Table 5.1 shows us that $\tilde{\mu}_{\max}$ is better for the global smoothers compared to the local smoothers, this is in agreement with the results in [10]. However, the maximum smoothing rate of

Smoother	Rada-Chen		Spencer-Chen	
	$\tilde{\mu}_{\max}$	$\tilde{\mu}_{\text{avg}}$	$\tilde{\mu}_{\max}$	$\tilde{\mu}_{\text{avg}}$
GSLINE-I	0.9997	0.4800	0.9990	0.4586
GSLINE-II	0.9997	0.3782	1.0000*	0.4893
GSLEX-I	0.9978	0.5807	0.9927	0.5269
GSLEX-II	1.0000*	0.5244	0.9996	0.5512
NEWT-I	0.9985	0.5642	0.9595	0.4839
NEWT-II	0.9999	0.5749	0.9950	0.5133

Table 5.1: Smoothers and the associated maximum and average smoothing rates for the Rada-Chen and Spencer-Chen models. * due to rounding.

all of the smoothers is bad and so they cannot be implemented in a successful multigrid scheme. We look to improve the maximum smoothing rate of one of the better schemes to obtain a smoother which can be implemented successfully. In the next section we will see that the problem is due to discontinuous coefficients in the numerical schemes, and so we look to [2, 50] which recommend the use of line smoothers rather than a pixel-by-pixel update approach. We, therefore, choose the GSLINE-I smoother and review its performance for the Rada-Chen model in detail to see if we can improve the maximum smoothing rate of 0.9997. The same approach will be applied to the Spencer-Chen model and the results will be quoted at the end of the next section.

Algorithm 1. In future discussions, when we compare other algorithms with Algorithm 1, this will be the FAS algorithm with GSLINE-I as smoother.

5.4. Non-linear multigrid Algorithm 2

We now consider how to improve the smoothers above to obtain a smoothing rate which is acceptable. This leads to our new hybrid smoothers and the resulting multigrid Algorithms 2 and 3.

5.4.1. An idea of adaptive iterative schemes

To gain more insight into the rates in Table 5.1, we first look only at those pixels (i, j) which have a large amplification factor. In Figure 5.1(a) we show the original image on

which the rate was measured and in Figure 5.1(b) the corresponding binary plot of those pixels where the amplification factor is above 0.6. We see that the smoother performs poorly at the edges of objects in the image, a phenomenon also observed in [32] where it was determined that the rate is poor due to the restriction and interpolation operators performing poorly at these points.

(a) (b) (c)

Figure 5.1: (a) Original image, (b) Pixels with a smoothing rate over 0.6 are indicated in white, (c) Pixels in white are those where one of the $A_{i,j}, B_{i,j}, C_{i,j}$ or $D_{i,j}$ values differs from the others by a factor of 50% or more.

There are two approaches that have been taken to address the poor smoothing rate at edges; the first is the use of adaptive high-order intergrid transfer operators [32] and the second is to apply extra smoothing steps at those edge points [15, 29, 32]. We prefer the second approach as the intergrid operators perform well generally and for ease of implementation in the current framework, the second approach is best. The conventional solution when doing extra smoothing steps would be to simply implement the *same* smoother many more times at those edge pixels to obtain a lower smoothing rate, however, we shall develop a *different* scheme to be used at these pixels which has an improved smoothing rate. In any case, we must first identify those pixels which contribute large amplification factors without needing to calculate $\hat{\mu}_{i,j}$ each time, which would be computationally expensive. In Table 5.2 we have selected the pixels in the image from Figure 5.1(a) which give 10 of the largest amplification factors and list the values of $A_{i,j}, B_{i,j}, C_{i,j}$ and $D_{i,j}$ at these pixels.

A pattern emerges that at these edge pixels (jumps) at least one of the values of $A_{i,j}, B_{i,j}, C_{i,j}$ and $D_{i,j}$ is significantly different to the others, Figure 5.1(c) shows those pixels where they differ by 50% (i.e. $\max(A_{i,j}, B_{i,j}, C_{i,j}, D_{i,j}) / \min(A_{i,j}, B_{i,j}, C_{i,j}, D_{i,j}) > 1.5$).

i	j	$\hat{\mu}_{i,j}$	$A_{i,j}$	$B_{i,j}$	$C_{i,j}$	$D_{i,j}$
46	23	0.9997	202	202	137391	35
45	23	0.9995	202	202	77788	35
25	23	0.9931	209	220	5545	36
42	112	0.9889	2263	1802	78959	842
44	82	0.9605	20	626	558	22

i	j	$\hat{\mu}_{i,j}$	$A_{i,j}$	$B_{i,j}$	$C_{i,j}$	$D_{i,j}$
44	112	0.9591	79987	6659	168919	6736
97	103	0.9551	3228	105968	72894	3203
80	60	0.9312	7937	424357	400718	27651
73	90	0.8756	29221	1426471	170469	21920
73	105	0.8750	321703	24343	242663	32126

Table 5.2: The pixels with 10 of the largest smoothing rates with the corresponding values of $A_{i,j}, B_{i,j}, C_{i,j}$ and $D_{i,j}$.

Definition 5.4.1.1. We can identify the edge pixels as those where at least one of $A_{i,j}, B_{i,j}, C_{i,j}$ or $D_{i,j}$ differs significantly from the others, this is precisely the set of jumps in the coefficients of (5.3), we denote this set by $\mathcal{D}$. For the set of pixels where $A_{i,j}, B_{i,j}, C_{i,j}$ or $D_{i,j}$ are relatively similar we denote it as $\Omega \backslash \mathcal{D}$.

We compare the maximum and average smoothing rates over $\mathcal{D}$ and $\Omega \backslash \mathcal{D}$ below:

Smoother	$\tilde{\mu}_{\max \mathcal{D}}$	$\tilde{\mu}_{\text{avg } \mathcal{D}}$	$\tilde{\mu}_{\max \Omega \backslash \mathcal{D}}$	$\tilde{\mu}_{\text{avg } \Omega \backslash \mathcal{D}}$
GSLINE-I	0.9997	0.5121	0.7705	0.4386

(5.17)

We see that the maximum amplification factor over $\Omega \backslash \mathcal{D}$ of 0.7705 would mean that the number of iterations required to reduce the high-frequency errors by 90% reduces from 7675 to 9. We now focus on reducing the amplification factor for the pixels of $\mathcal{D}$.

Classifying the jumps. There are 14 possible cases to consider where one of the coefficients $A_{i,j}, B_{i,j}, C_{i,j}$ or $D_{i,j}$ is relatively larger (L) or smaller (S) than the others, these are all shown below:

Case #	$A_{i,j}$	$B_{i,j}$	$C_{i,j}$	$D_{i,j}$
1	S	L	L	S
2	S	L	S	L
3	L	S	L	S
4	L	S	S	L
5	L	L	S	S
6	S	S	L	L
7	L	S	S	S

Case #	$A_{i,j}$	$B_{i,j}$	$C_{i,j}$	$D_{i,j}$
8	S	S	L	S
9	S	L	S	S
10	S	S	S	L
11	L	L	S	L
12	L	S	L	L
13	L	L	L	S
14	S	L	L	L

(5.18)

We can now label each pixel in $\mathcal{D}$ as one of the cases from 1 to 14. The choice of label

L or S for a coefficient will be dependent on the coefficients at each pixel. Typically, if the largest coefficient is 50% larger than the smallest we group the coefficients as large or small by K-means or some other classification method. For a pixel in $\mathcal{D}$, we now look to adapt the iterative scheme (5.12) for each of these cases to give a scheme which has a better smoothing rate than implementing GSLINE-I directly. In the interests of brevity, we consider Case 1 in detail and will generalise the results to other cases next.

5.4.1.1 An adapted iterative scheme and its LFA form

Our aim is to propose a new iteration scheme which leads to a smaller smoothing rate by the LFA. For Case 1 pixels, $A_{i,j}$ and $D_{i,j}$ are relatively small and $B_{i,j}$ and $C_{i,j}$ are relatively large. We can rewrite (5.12) as

$$B_{i,j}\phi_{i-1,j} + C_{i,j}\phi_{i,j+1} - S_{i,j}\phi_{i,j} = f_{i,j} - A_{i,j}\phi_{i+1,j} - D_{i,j}\phi_{i,j-1},$$

by moving the small terms to the right-hand side. We now look to solve $\phi_{i-1,j}, \phi_{i,j+1}$ and $\phi_{i,j}$ as a coupled system. We can rewrite this scheme, with the iteration number indicated, as

$$B_{i,j}\phi_{i-1,j}^{(k+1)} + C_{i,j}\phi_{i,j+1}^{(k+1)} - S_{i,j}\phi_{i,j}^{(k+1)} = f_{i,j} - A_{i,j}\phi_{i+1,j}^{(k)} - D_{i,j}\phi_{i,j-1}^{(k)}. \tag{5.19}$$

The amplification factor for such a scheme is

$$\hat{\mu}_{i,j} = \max_{\theta_1,\theta_2} \mu(\theta_1,\theta_2) = \max_{\alpha_1,\alpha_2} \frac{|A_{i,j}e^{\mathbf{i}\alpha_1} + D_{i,j}e^{-\mathbf{i}\alpha_2}|}{|S_{i,j} - B_{i,j}e^{-\mathbf{i}\alpha_1} - C_{i,j}e^{\mathbf{i}\alpha_2}|}, \tag{5.20}$$

derived as in §5.3.4. In fact, we see the following improvements to the maximum and average smoothing rates for all of the Case 1 pixels by using the adapted iterative scheme (5.19) rather than the GSLINE-I smoother in (5.10)

$$\tilde{\mu}_{\max} = 0.9863,\ \tilde{\mu}_{\text{avg}} = 0.7174 \quad \Longrightarrow \quad \tilde{\mu}_{\max} = 0.7324,\ \tilde{\mu}_{\text{avg}} = 0.3013$$

Reducing the smoothing rate from 0.9863 to 0.7324 is dramatic; exemplified by the fact that to reduce high-frequency errors by 90% for Case 1 pixels with GSLINE-I we would have required 167 iterations but now we need just 8. Hence, now we know that the

scheme (5.19) gives us a better smoothing rate than GSLINE-I at these pixels.

5.4.1.2 Adapted schemes for all cases and their rates by LFA

Using the central idea of lagging the small terms in (5.18) (between 1 and 3 terms), we can derive adapted schemes for all cases in the same manner as for Case 1 previously. In Table 5.3 we give the comparison of the maximum smoothing rate of GSLINE-I, μ_{GSLINE}, with the maximum smoothing rate of the adapted schemes $\mu_{adapted_1}$.

Case #	μ_{GSLINE}	$\mu_{adapted_1}$		Case #	μ_{GSLINE}	$\mu_{adapted_1}$	
1	0.9863	0.7324	♢	8	0.9997	0.9569	♢
2	0.6259	0.8515	♢	9	0.9481	0.9426	♢
3	0.9900	0.7418	♢	10	0.8935	0.9640	♢
4	0.6408	0.7415	♢	11	0.2693	0.2693	♠
5	0.7105	1.0000	□	12	0.7729	0.2663	♠
6	0.9524	1.0000	□	13	0.9865	0.2704	♠
7	0.9592	0.9536	♢	14	0.5993	0.2706	♠

Table 5.3: Comparison of the maximum amplification factors using GSLINE-I and the adapted iterative schemes for each case. The □-cases are the decoupled cases which give a rate of precisely 1, as remarked, the ♢-cases have minor or no improvement in the smoothing rate and the ♠-cases have a good final rate.

The results from Table 5.3 fall into 3 categories:

♠-cases, where only one term is lagged and the improvements are remarkable. This gives a promising indication that the lagging of particular terms in certain cases can improve the smoothing rate. This motivates our next step.

♢-cases, where either 2 or 3 terms are lagged. We see either only a minor improvement to an already high rate or the rate has actually worsened.

□-cases, where 2 terms are lagged and we see the worst results: a smoothing rate of 1.0000 is attained for cases 5, 6 in Table 5.3. Below we prove analytically that for Case 6 pixels the smoothing rate when using the adapted scheme will always be precisely 1.

Case 6 pixels have the LFA form $\hat{\mu}_{i,j} = \max_{\alpha_1,\alpha_2} \frac{|A_{i,j}e^{\mathbf{i}\alpha_1}+B_{i,j}e^{-\mathbf{i}\alpha_1}|}{|S_{i,j}-C_{i,j}e^{\mathbf{i}\alpha_2}-D_{i,j}e^{-\mathbf{i}\alpha_2}|}$, and we see a decoupling in the maximisation with respect to α_1 and α_2 which allows us to

rewrite this as

$$\begin{aligned}\hat{\mu}_{i,j} &= \frac{\max\limits_{\alpha_1}\left|A_{i,j}e^{\mathbf{i}\alpha_1}+B_{i,j}e^{-\mathbf{i}\alpha_1}\right|}{\min\limits_{\alpha_2}\left|S_{i,j}-C_{i,j}e^{\mathbf{i}\alpha_2}-D_{i,j}e^{-\mathbf{i}\alpha_2}\right|}\\ &= \frac{\max\limits_{\alpha_1}\left|(A_{i,j}+B_{i,j})\cos(\alpha_1)+\mathbf{i}(A_{i,j}-B_{i,j})\sin(\alpha_1)\right|}{\min\limits_{\alpha_2}\left|\left[A_{i,j}+B_{i,j}+C_{i,j}(1-\cos(\alpha_2))+D_{i,j}(1-\cos(\alpha_2))\right]+\mathbf{i}(C_{i,j}-D_{i,j})\sin(\alpha_2)\right|}\\ &= \frac{\sqrt{\max\limits_{\alpha_1}\left[A_{i,j}^2+B_{i,j}^2+2A_{i,j}B_{i,j}cos(2\alpha_1)\right]}}{\sqrt{\min\limits_{\alpha_2}\left[\left[A_{i,j}+B_{i,j}+C_{i,j}(1-\cos(\alpha_2))+D_{i,j}(1-\cos(\alpha_2))\right]^2+(C_{i,j}-D_{i,j})^2\sin(\alpha_2)^2\right]}}\\ &= \frac{(A_{i,j}+B_{i,j})^2}{(A_{i,j}+B_{i,j})^2}=1,\end{aligned}$$

attained at $(\alpha_1,\alpha_2)=(-\pi,0)\in[-\pi,\pi)^2\backslash[-\frac{\pi}{2},\frac{\pi}{2})^2$. Similarly we have $\hat{\mu}_{i,j}=1$ for Case 5 too.

We claim that it is necessary to have both of α_1 and α_2 in the numerator or denominator of the LFA formulation to ensure a low smoothing rate. We note that for Cases 5 and 6 this is not the case.

We now focus on improving the $\diamondsuit$-cases and the Case 8 in particular and its LFA to motivate us on how to proceed i.e. to see whether an alternative adaptation to the iterative scheme gives a better smoothing rate. The results apply to $\square$-cases also.

Improving the adapted scheme for Case 8. A pixel which is labelled as Case 8 is one where $A_{i,j}, B_{i,j}, D_{i,j}$ are relatively small and $C_{i,j}$ is relatively large. Using the previous method we would devise a scheme where the terms with coefficients $A_{i,j}, B_{i,j}, D_{i,j}$ would be lagged at time step k and the term with coefficient $C_{i,j}$ would be updated to time step $k+1$. We pick the particular Case 8 pixel from Table 5.2 which has the worst smoothing rate and in Figure 5.2 we look at the smoothing rate for the scheme (5.12) with different coefficients lagged.

This shows that the best rate is achieved when just the smallest of the coefficients ($D_{i,j}$) is lagged. Even the lagging of two of the smallest coefficients gives an improvement on lagging all three. This gives some indication that the smoothing rate is best when the smallest coefficient is lagged and this has proven to be the case in every one of the many

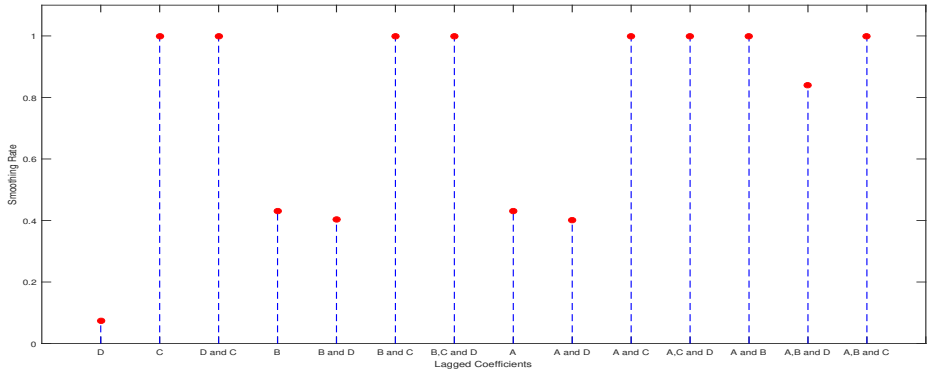


Figure 5.2: Comparison of the smoothing rate for the Case 8 pixel with the worst smoothing rate when different coefficient terms are lagged. In this case, $A_{i,j} = 202, B_{i,j} = 202, C_{i,j} = 137391$ and $D_{i,j} = 35$ (Table 5.2).

examples which the authors have tried. It would be an interesting piece of future work to prove that this must be true analytically.

Remark 5.4.1.2. To prove this analytically an approach taken by the author, but abandoned due to time constraints, was to compare the LFA forms for lagging the smallest coefficient with the LFA forms for all other such laggings. The aim being to prove that lagging the smallest coefficient gives a smaller rate than all others. For example, suppose $A_{i,j}$ is the smallest coefficient and we focus on lagging $A_{i,j}$ and lagging both $A_{i,j}$ and $C_{i,j}$. Then we are interested in comparing the maxima of

$$\frac{|A_{i,j}e^{\mathbf{i}\alpha_1} + C_{i,j}e^{\mathbf{i}\alpha_2}|}{|S_{i,j} - B_{i,j}e^{-\mathbf{i}\alpha_1} - D_{i,j}e^{-\mathbf{i}\alpha_2}|} \quad \text{and} \quad \frac{|A_{i,j}|}{|S_{i,j} - B_{i,j}e^{-\mathbf{i}\alpha_1} - C_{i,j}e^{\mathbf{i}\alpha_2} - D_{i,j}e^{-\mathbf{i}\alpha_2}|}$$

Denote these respective sets of maxima values by $\boldsymbol{\mu}_{AC}$ and $\boldsymbol{\mu}_A$. Then we aim to prove that the largest value of $\boldsymbol{\mu}_{AC}$ is always larger than the largest value of $\boldsymbol{\mu}_A$. Isolating the extrema is the first step and is accomplished by finding the partial derivatives of the above LFA forms for α_1 and α_2. Then methodically, one can compare the maxima. This is terribly time consuming so a more efficient method would be preferred.

Hence we propose to lag just the smallest of the coefficients in a modified scheme for all cases.

5.4.1.3 Improved adapted schemes for all cases

We re-consider the $\diamond$ and $\square$-cases which have more than one relatively small coefficient. Lagging only the smallest coefficient, the LFA forms simplify to those of Cases 11–14 and we expect major improvements. In Table 5.4 we compare the maximum smoothing rate of GSLINE-I, μ_{GSLINE}, for these cases with the maximum smoothing rate of an improved, adapted iterative scheme which lags only the smallest coefficient $\mu_{adapted_2}$.

Case #	μ_{GSLINE}	$\mu_{adapted_2}$	Case #	μ_{GSLINE}	$\mu_{adapted_2}$
1	0.9863	0.4467	8	0.9997	0.4779
2	0.6259	0.4398	9	0.9481	0.4716
3	0.9900	0.4280	10	0.8935	0.4749
4	0.6408	0.4468	11	0.2693	0.2693
5	0.7105	0.4659	12	0.7729	0.2663
6	0.9524	0.4547	13	0.9865	0.2704
7	0.9592	0.4789	14	0.5993	0.2706

Table 5.4: Comparison of the maximum amplification factors using GSLINE-I and the adapted iterative schemes for each case with just the smallest coefficient term lagged.

As expected, there is a significant improvement in the smoothing rate in all cases when we lag just the smallest coefficient, it also makes implementation faster as we now consider just 4 cases of possible lagged coefficients rather than 14 and therefore have only 4 iterative schemes to consider. Taking our guidance from these results, we propose two hybrid smoothers which both perform standard smoothing iterations on pixels of $\Omega \backslash \mathcal{D}$ and perform non-standard adapted iterative schemes on the pixels in $\mathcal{D}$.

Based on the above pixel-wise motivating tests, we now present two iterative strategies for our new smoothers. The first smoother is natural: for each pixel (i, j), in $\mathcal{D}$, all of the directly connected neighbouring pixels are collectively updated except the term with the smallest coefficient. That is, Hybrid Smoother 1 uses block structure Vanka-type smoothing schemes [141, 161] to update the pixels in $\mathcal{D}$. The potential drawback is that previously updated pixels may enter to the next group of (potentially multiple) updates, making subsequent analysis intractable. Hence our second smoother, denoted by 'Hybrid Smoother 2', incorporates partial line smoothing operations at pixels in $\mathcal{D}$ and only pixels that are the same line as (i, j) are updated. This line by line approach facilitates subsequent analysis.

5.4.2. Hybrid Smoother 1

Our first hybrid smoother updates blocks of pixels at each update, these blocks may overlap. This is an overlapping block smoother of Vanka-type [141, 161]. Once again we start with the set $\mathcal{D}$ of pixels with jumping coefficients. For brevity, we will detail the derivation of the iterative scheme for pixels in $\mathcal{D}$ for which $A_{i,j}$ is the smallest. We will then state the schemes for the other laggings (derived in the same manner).

$\boldsymbol{A_{i,j}}$ **lagged.** The lagging of coefficient $A_{i,j}$ in equation (5.12) gives rise to the iterative scheme

$$A_{i,j}\phi_{i+1,j}^{(k)} + B_{i,j}\phi_{i-1,j}^{(k+1)} + C_{i,j}\phi_{i,j+1}^{(k+1)} + D_{i,j}\phi_{i,j-1}^{(k+1)} - S_{i,j}\phi_{i,j}^{(k+1)} = f_{i,j}, \tag{5.21}$$

We are solving for $\phi_{i-1,j}, \phi_{i,j+1}, \phi_{i,j-1}$ and $\phi_{i,j}$ simultaneously and as we have only one equation, we need three more. We get these by considering (5.12) at the pixels $(i-1, j)$ and $(i, j+1)$ and $(i, j-1)$, which gives us the three equations

$$\begin{aligned}
B_{i,j}\phi_{i,j} - S_{i-1,j}\phi_{i-1,j} &= f_{i-1,j} - B_{i-1,j}\phi_{i-2,j} - C_{i-1,j}\phi_{i-1,j+1} - D_{i-1,j}\phi_{i-1,j-1},\\
C_{i,j}\phi_{i,j} - S_{i,j+1}\phi_{i,j+1} &= f_{i,j+1} - A_{i,j+1}\phi_{i+1,j+1} - B_{i,j+1}\phi_{i-1,j+1} - C_{i,j+1}\phi_{i,j+2},\\
D_{i,j}\phi_{i,j} - S_{i,j-1}\phi_{i,j-1} &= f_{i,j-1} - A_{i,j-1}\phi_{i+1,j-1} - B_{i,j-1}\phi_{i-1,j-1} - D_{i,j-1}\phi_{i,j-2},
\end{aligned}$$

which have been rearranged to have the $\phi_{i-1,j}, \phi_{i,j+1}, \phi_{i,j-1}$ and $\phi_{i,j}$ terms on the left-hand side. So, using these along with (5.21) we obtain the system (5.22).

Scheme with $\boldsymbol{A_{i,j}}$ lagged:

$$\begin{pmatrix} -S_{i,j} & B_{i,j} & C_{i,j} & D_{i,j} \\ B_{i,j} & -S_{i-1,j} & 0 & 0 \\ C_{i,j} & 0 & -S_{i,j+1} & 0 \\ D_{i,j} & 0 & 0 & -S_{i,j-1} \end{pmatrix} \cdot \begin{pmatrix} \phi_{i,j} \\ \phi_{i-1,j} \\ \phi_{i,j+1} \\ \phi_{i,j-1} \end{pmatrix} = \begin{pmatrix} f_{i,j} - A_{i,j}\phi_{i+1,j} \\ f_{i-1,j} - C_{i-1,j}\phi_{i-1,j+1} - D_{i-1,j}\phi_{i-1,j-1} - B_{i-1,j}\phi_{i-2,j} \\ f_{i,j+1} - A_{i,j+1}\phi_{i+1,j+1} - B_{i,j+1}\phi_{i-1,j+1} - C_{i,j+1}\phi_{i,j+2} \\ f_{i,j-1} - A_{i,j-1}\phi_{i+1,j-1} - B_{i,j-1}\phi_{i-1,j-1} - D_{i,j-1}\phi_{i,j-2} \end{pmatrix}. \tag{5.22}$$

This system is strictly diagonally dominant and follows the guidance in [158] that collective update schemes are better for jumping coefficients. This system also has an arrow structure in the matrix and can be solved very quickly (in 24 operations).

5.4.2.1 The adapted iterative schemes for other cases

Below are the adapted iterative schemes for the cases when $B_{i,j}, C_{i,j}$ or $D_{i,j}$ are lagged, derived in the same manner as previously when $A_{i,j}$ was lagged.

Scheme with $B_{i,j}$ lagged:

$$\begin{pmatrix} -S_{i,j} & A_{i,j} & C_{i,j} & D_{i,j} \\ A_{i,j} & -S_{i+1,j} & 0 & 0 \\ C_{i,j} & 0 & -S_{i,j+1} & 0 \\ D_{i,j} & 0 & 0 & -S_{i,j-1} \end{pmatrix} \cdot \begin{pmatrix} \phi_{i,j} \\ \phi_{i+1,j} \\ \phi_{i,j+1} \\ \phi_{i,j-1} \end{pmatrix} = \begin{pmatrix} f_{i,j} - B_{i,j}\phi_{i-1,j} \\ f_{i+1,j} - C_{i+1,j}\phi_{i+1,j+1} - D_{i+1,j}\phi_{i+1,j-1} - A_{i+1,j}\phi_{i+2,j} \\ f_{i,j+1} - A_{i,j+1}\phi_{i+1,j+1} - B_{i,j+1}\phi_{i-1,j+1} - C_{i,j+1}\phi_{i,j+2} \\ f_{i,j-1} - A_{i,j-1}\phi_{i+1,j-1} - B_{i,j-1}\phi_{i-1,j-1} - D_{i,j-1}\phi_{i,j-2} \end{pmatrix}. \quad (5.23)$$

Scheme with $C_{i,j}$ lagged:

$$\begin{pmatrix} -S_{i,j} & A_{i,j} & B_{i,j} & D_{i,j} \\ A_{i,j} & -S_{i+1,j} & 0 & 0 \\ B_{i,j} & 0 & -S_{i-1,j} & 0 \\ D_{i,j} & 0 & 0 & -S_{i,j-1} \end{pmatrix} \cdot \begin{pmatrix} \phi_{i,j} \\ \phi_{i+1,j} \\ \phi_{i-1,j} \\ \phi_{i,j-1} \end{pmatrix} = \begin{pmatrix} f_{i,j} - C_{i,j}\phi_{i,j+1} \\ f_{i+1,j} - C_{i+1,j}\phi_{i+1,j+1} - D_{i+1,j}\phi_{i+1,j-1} - A_{i+1,j}\phi_{i+2,j} \\ f_{i-1,j} - C_{i-1,j}\phi_{i-1,j+1} - D_{i-1,j}\phi_{i-1,j-1} - B_{i-1,j}\phi_{i-2,j} \\ f_{i,j-1} - A_{i,j-1}\phi_{i+1,j-1} - B_{i,j-1}\phi_{i-1,j-1} - D_{i,j-1}\phi_{i,j-2} \end{pmatrix}. \quad (5.24)$$

Scheme with $D_{i,j}$ lagged:

$$\begin{pmatrix} -S_{i,j} & A_{i,j} & B_{i,j} & C_{i,j} \\ A_{i,j} & -S_{i+1,j} & 0 & 0 \\ B_{i,j} & 0 & -S_{i-1,j} & 0 \\ C_{i,j} & 0 & 0 & -S_{i,j+1} \end{pmatrix} \cdot \begin{pmatrix} \phi_{i,j} \\ \phi_{i+1,j} \\ \phi_{i-1,j} \\ \phi_{i,j+1} \end{pmatrix} = \begin{pmatrix} f_{i,j} - D_{i,j}\phi_{i,j-1} \\ f_{i+1,j} - C_{i+1,j}\phi_{i+1,j+1} - D_{i+1,j}\phi_{i+1,j-1} - A_{i+1,j}\phi_{i+2,j} \\ f_{i-1,j} - C_{i-1,j}\phi_{i-1,j+1} - D_{i-1,j}\phi_{i-1,j-1} - B_{i-1,j}\phi_{i-2,j} \\ f_{i,j+1} - A_{i,j+1}\phi_{i+1,j+1} - B_{i,j+1}\phi_{i-1,j+1} - C_{i,j+1}\phi_{i,j+2} \end{pmatrix}. \quad (5.25)$$

5.4.2.2 Implementing Hybrid Smoother 1

To minimise grid sweeps and ensure that all pixels are covered, we use the following pseudo-algorithm for Hybrid Smoother 2:

I Perform GSLINE-I on all lines in the image.

II For each pixel in $\mathcal{D}$, perform the appropriate scheme of (5.22)–(5.25).

We justify the choice of GSLINE-I in step **I** as it is the recommended smoothing scheme for a problem with jump coefficients [158]. Note that the schemes in **II** can overlap the same pixels several times due to the collective updates.

Algorithm 2. In future discussion, when we use the Hybrid Smoother 1 in the Full Approximation Scheme, we will call this Algorithm 2.

5.4.3. Hybrid Smoother 2

Our second hybrid smoother first groups pixels in $\mathcal{D}$ by whether $A_{i,j}$, $B_{i,j}$, $C_{i,j}$ or $D_{i,j}$ are the smallest and then by the line they are on. We then perform partial line updates on these groups for $A_{i,j}$, $B_{i,j}$, $C_{i,j}$ or $D_{i,j}$ in sequence along with individual pixel updates on the other pixels, this avoids the overlap encountered in Hybrid Smoother 1. We note that for pixels in $\Omega \backslash \mathcal{D}$ the LFA tells us that the smoothing rate is acceptable (maximum 0.7705) and therefore we design a smoother which performs cheap GSLEX-I iterations at the pixels of $\Omega \backslash \mathcal{D}$ and performs the lagged scheme on the other pixels. We focus initially on how we propose implementing this for the pixels in $\mathcal{D}$ with $A_{i,j}$ lagged and then we generalise the idea to the laggings of $B_{i,j}, C_{i,j}$ and $D_{i,j}$.

Scheme with $A_{i,j}$ lagged. Suppose we focus on a pixel $(i,j) \in \mathcal{D}$ which has coefficient $A_{i,j}$ the smallest. If we lag the $A_{i,j}$ the smoothing rate at this pixel is

$$\hat{\mu}_{i,j} = \max_{(\alpha_1,\alpha_2)\in[-\pi,\pi)^2 \backslash [-\frac{\pi}{2},\frac{\pi}{2})^2} \left| \frac{A_{i,j}e^{\mathbf{i}\alpha_1}}{B_{i,j}e^{-\mathbf{i}\alpha_1} + C_{i,j}e^{\mathbf{i}\alpha_2} + D_{i,j}e^{-\mathbf{i}\alpha_2} - S_{i,j}} \right|$$

which is precisely the smoothing rate for a line smoother updating from the top row to the bottom row. In the majority of cases, if pixel $A_{i,j}$ is the smallest, we find that many adjacent pixels on that line also have $A_{i,\cdot}$ the smallest. So we can perform a partial line smoothing on these pixels.

In this new strategy, the only technical issue to address is that, at a pixel (i,j) in set $\mathcal{D}$, the lagged coefficient (here $A_{i,j}$) must be a previously updated pixel in this iteration otherwise we cannot avoid multiple updates (as with Hybrid Smoother 1) within one smoothing iteration. Our proposed solution is to view a group of adjacent pixels in set $\mathcal{D}$ whose smallest coefficient is $A_{i,j}$ (shown as starred pixels in Figure 5.3) and sit on a line as a superpixel and to update together with their $A_{i,j}$ terms lagged. If the superpixel is comprised of a single pixel, we set its immediate neighbour pixel (here $(i,j+1)$) as a starred pixel so the group is of size 2. All other pixels in set $\mathcal{D}$ (without smallest coefficient $A_{i,j}$) and those not in $\mathcal{D}$ are treated as normal pixels (non-starred) and are relaxed by the GSLEX-1 formula. Hence in each smoothing step, starred and non-starred pixels are only updated once.

In Figure 5.3 we illustrate how this proposed algorithm would update the pixels, steps **I–**

VI represent one iteration of the smoother on the 5×5 grid. The starred pixels represent those pixels which have $A_{i,j}$ the smallest. The algorithm proceeds as follows:

I We identify the pixels in $\mathcal{D}$ which have $A_{i,j}$ the smallest (indicated by a star).

II Perform GSLEX-I on all non-starred pixels.

III Collective partial line update on adjacent starred pixels.

IV Perform GSLEX-I again on all non-starred pixels.

V If a single starred pixel is found, update collectively with the immediate neighbour.

VI Perform GSLEX-I again on all non-starred pixels.

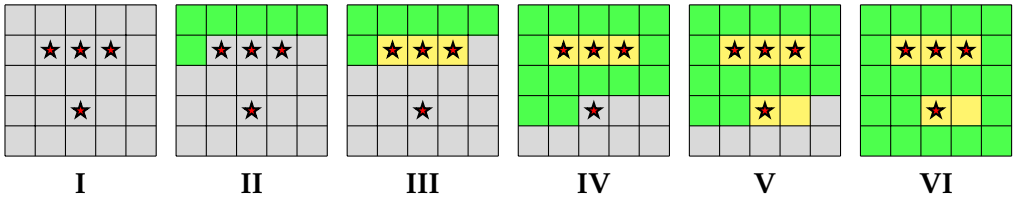


Figure 5.3: Illustration of the hybrid algorithm for a pixel grid. Each image represents one step of the algorithm, grey cells are yet to be updated. The star pixels are pixels in $\mathcal{D}$ with $A_{i,j}$ smallest. Green represents the update by GSLEX-I and the yellow pixels are the partial line smoothing updates.

5.4.3.1 The adapted iterative schemes for other cases

We previously focused on the case for $A_{i,j}$ being lagged and now discuss other components of our iterative scheme to cover the cases of $B_{i,j}, C_{i,j}$ and $D_{i,j}$ being lagged.

Crucially, to ensure that the scheme agrees with the LFA we must change the direction of update between the schemes for updating $A_{i,j}$, $B_{i,j}$, $C_{i,j}$ and $D_{i,j}$. For example, if we are lagging $B_{i,j}$ pixels we must update from the bottom-right corner to the top-left moving along rows right to left and from the bottom row to the top row. In Figure 5.4 we show the order in which the pixels should be updated for each lagging.

These sweeps in other directions are required to help those pixels in $\mathcal{D}$ that were treated as non-starred pixels due to their smallest coefficients not being considered in the other sweeps. That is to say, each of 4 sweeps takes care of one type of alignment of the

smallest coefficients (of course there are no other directions to consider). Consequently, after all 4 sweeps, the compounded smoothing rate at each pixel is small because we have ensured that one of the four multiplying factors is small while the other three are no more than 1.

The broad algorithm (**I–VI**) is the same in these cases as for the case of $A_{i,j}$ lagged; we identify the pixels which are of that case, perform GSLEX-I on all others and partial line updates on identified pixels.

Hybrid Smoother 2 performs 4 sweeps of the grid, each repeating the above **I–V** and differing only in update order and assignment of starred pixels. In Figure 5.4 we display the order in which the pixels and superpixels should be updated for each lagging.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

$\boldsymbol{A_{i,j}}$ **Lagged**

25	24	23	22	21
20	19	18	17	16
15	14	13	12	11
10	9	8	7	6
5	4	3	2	1

$\boldsymbol{B_{i,j}}$ **Lagged**

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

$\boldsymbol{C_{i,j}}$ **Lagged**

25	20	15	10	5
24	19	14	9	4
23	18	13	8	3
22	17	12	7	2
21	16	11	6	1

$\boldsymbol{D_{i,j}}$ **Lagged**

Figure 5.4: Illustration of the hybrid algorithm for a pixel grid. The star pixels are pixels in $\mathcal{D}$ with $A_{i,j}$ smallest. Green represents the update by GSLEX-I and the yellow pixels are the partial line smoothing updates.

5.4.3.2 Implementing Hybrid Smoother 2

To ensure all laggings are considered, we sweep for $A_{i,j}$, $B_{i,j}$, $C_{i,j}$ and $D_{i,j}$ in this order, performing steps (**I–VI**) on each sweep. These schemes are performed on all pixels in $\mathcal{D}$ and we see from Table 5.4 that the maximum smoothing rate over $\mathcal{D}$ falls from 0.9997 to 0.4789. Therefore to reduce high-frequency errors by 90%, with GSLINE-I this would have needed 7675 iterations but with the adapted iterative schemes we need only 4.

To ensure that all cases are considered, we design a hybrid smoother for which one outer iteration includes four sweeps of the image domain. In the first sweep, we lag $A_{i,j}$, then in the second $B_{i,j}$ and so on. We note, for example, that in the sweep with $A_{i,j}$ lagged, then the pixels with coefficient $B_{i,j}$ smallest have a poor smoothing rate, however, on

the $B_{i,j}$ sweep the rate is good for these pixels and poor for those where we have $A_{i,j}$ smallest. However, as the effects compound multiplicatively, after each outer iteration, the smoothing rate at pixels in $\mathcal{D}$ is good and for $\Omega\backslash\mathcal{D}$ is also good as these have had 4 GSLEX-I iterations.

We now consider the smoothing rates we can attain with this smoother. Firstly, for the Rada-Chen model [135], using (5.17) we see that the maximum smoothing rate in each outer iteration of the smoother on $\Omega\backslash\mathcal{D}$ is approximately $0.7705^4 = 0.3524$. By performing the adapted iterative schemes on $\mathcal{D}$ we have a maximum smoothing rate of 0.4789 (Table 5.4) in a single sweep. We know that the rate for GSLEX-I is poor for these pixels in $\mathcal{D}$ (close to 1) so the main reduction in error occurs when we perform the adapted scheme with the appropriate lagging. Therefore the maximum smoothing rate in one outer iteration of the smoother is approximately 0.4789, which is very good. One consideration we must make is that the domain is covered 4 times in each outer iteration, which could be computationally intensive for a large number of smoothing steps. Typically we find that for non-linear problems the number of overall sweeps of the grid is around 10-20 (see, for example, [32, 176]) for the smoother, therefore we suggest 2 outer iterations (8 grid sweeps) which gives an impressive smoothing rate and is acceptable computationally.

Adaptive iterative schemes applied to the Spencer-Chen model [153]. We applied Hybrid Smoother 2 to the Spencer-Chen model. In this case, using just GSLINE-I we have a maximum smoothing rate of 0.9990 but using the new smoother, the maximum smoothing rate falls to 0.5032. Therefore, to reduce errors by 90% we need 4 iterations rather than 2302. This is a further indication that the technique of using the partial line smoothers at the pixels with jumps in the coefficients is a good way to reduce the maximum smoothing rate of the smoother and the idea transfers to other models.

Improved smoothing rates for other images. We now show how the maximum smoothing rate for Hybrid Smoother 2 is smaller than GSLINE-I for several images with different levels of Gaussian noise. We compare to GSLINE-I as this is the recommended standard smoother for problems with jumping coefficients. We denote the corresponding maximum smoothing rates as $\mu_{GSLINE-I}$ and $\mu_{GSHYBRID}$ respectively. Results obtained previously are just for the clean image in Figure 5.1(a). Here we compare the smoothing rates for noisy versions of this image and also of those in Figure 5.5.

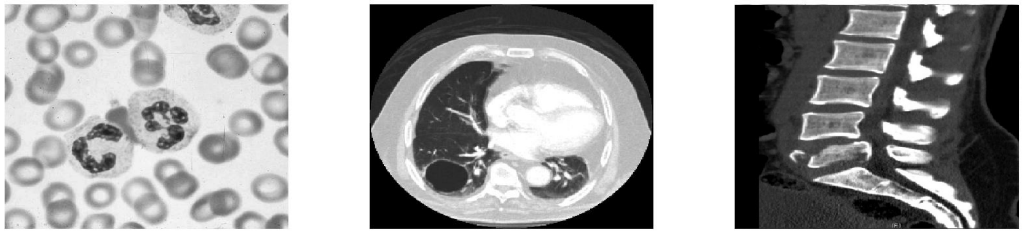
Image	$\mu_{GSLINE-I}$	μ_{HYBRID}
Figure 5.1(a) + 1% Noise	0.9743	0.4891
Figure 5.1(a) + 5% Noise	0.9851	0.4815
Problem 1	0.9960	0.4532
Problem 1 + 1% Noise	0.9900	0.4749
Problem 1 + 5% Noise	0.9991	0.4789
Problem 2	0.9999	0.4736
Problem 2 + 1% Noise	0.9988	0.4886
Problem 2 + 5% Noise	0.9934	0.4518
Problem 3	0.9999	0.4829
Problem 3 + 1% Noise	0.9999	0.4863
Problem 3 + 5% Noise	0.9999	0.4841

Table 5.5: Comparison of the maximum smoothing rates for GSLINE-I and Hybrid Smoother 2 for various images.

Algorithm 3. In future discussion, we refer to the Full Approximation Scheme using Hybrid Smoother 2 as Algorithm 3.

5.5. Numerical Experiments

In this section, we show two types of numerical experiments: comparisons with the current best methods and analysis of the complexity of Algorithms 2 and 3. Results have been obtained for many artificial and real images but we restrict to the images shown in Figure 5.5. We show real images as these are of most interest for the application of selective segmentation. The Rada-Chen and Spencer-Chen models we look at are



Problem 1 Problem 2 Problem 3

Figure 5.5: The test images used in this section for the experiments.

non-convex and we, therefore, need the initialisation to be close to the final solution. Thankfully this can be achieved by setting the initial contour as the boundary of the polygon formed from the user selected points in $\mathcal{S}$. For examples of such user-defined points, see Figure 5.7.

Parameter Choices. The values of c_1 and c_2, being the average intensities inside and outside of the contour, are updated at the end of each multigrid iteration - the initial values are set to the average inside and outside the initial contour. We fix $\mu = 1/2$, $\lambda_1 = \lambda_2 = 10^{-4}$, $\nu = 1$ (for the Rada-Chen model) and $\theta = 1$ (for the Spencer-Chen model). In all experiments we use a V-cycle, i.e. fix $\gamma = 1$.

Number of Smoothing Steps. To decide how many smoothing steps were required in Algorithms 1, 2 and 3, we performed experiments to see how the number of smoothing steps impacted the number of multigrid cycles for convergence. As the number of smoothing steps increases, the number of cycles decreases and plateaus. We fix the number of smoothing steps for each algorithm as the number required for the number of multigrid cycles to the first plateau. In Figure 5.6 we demonstrate how the number of multigrid cycles required for convergence changes with the number of smoothing steps and how we choose the optimal number of pre- and post-smoothing steps (ν_1 and ν_2). In all tests, we use 100 iterations of the exact solver (AOS) on the coarsest level. Using this

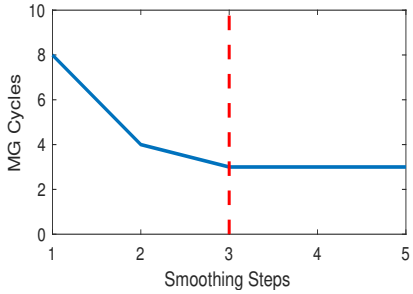


Figure 5.6: The number of smoothing steps plotted against the number of multigrid cycles required to achieve convergence for Algorithm 3 on Problem 1. Guided by this, we choose 3 smoothing steps as the gain plateau's at this point.

technique, we fix the smoothing steps for Algorithms 1, 2 and 3 as $\nu_1 = \nu_2 = 5$, 3 and 3 respectively.

5.5.1. Comparison of Algorithm 2 and Algorithm 3 with AOS

In this section, we compare the speed of the proposed Algorithms 2 and 3 with AOS. We use the image from Problem 1 and scale this to different resolutions. The methods both use the standard stopping criteria $\frac{||\phi^{(k+1)}-\phi^{(k)}||_2}{||\phi^{(k)}||_2} < \eta$, where η is a small tolerance parameter. In Table 5.6 we see that Algorithm 3 is faster to reach the stopping criteria (with $\eta = 10^{-4}$) than Algorithm 2 and that both are faster than AOS for all but the smallest resolution image. We see that as the image size grows larger, performance is significantly better. One key aspect of Algorithms 2 and 3 is that we have the expected ratio for an $\mathcal{O}(N)$ method (in 2D) of 4 and hence an optimal complexity multigrid method. We also see that the multigrid method has a stable number of overall iterations, whereas, with the AOS method, the iteration number grows as the image size grows. Finally, we see that, although it converges faster overall, the cost per MG cycle is larger for Algorithm 3 than 2. This is due to a higher number of grid sweeps being required in the smoothing steps, however, we believe that with improved and optimised coding of the smoother the performance of Algorithm 3 can be increased to achieve far faster convergence than that of Algorithm 2.

Image size	Number of Unknowns, N	AOS		Algorithm 2			Algorithm 3		
		Iter	CPU Time (s)	Iter	CPU Time (s)	CPU Ratio	Iter	CPU Time (s)	CPU Ratio
256×256	65536	32	3.2	4	3.1	-	4	8.8	-
512×512	262144	39	17.3	5	11.6	3.7	3	15.0	1.7
1024×1024	1048576	48	123.5	5	44.0	3.8	3	43.8	2.9
2048×2048	4194304	60	759.2	5	174.2	4.0	3	174.1	4.0
4096×4096	16777216	75	8632.4	5	725.9	4.2	3	688.2	4.0
8192×8192	67108864	*	*	5	2952.2	4.1	3	2766.9	4.0

Table 5.6: For an image of size $N = m \times n$, we show a comparison of the number of iterations and the associated CPU times to achieve the same results for the Rada-Chen model for AOS and Algorithms 2 and 3. '*' indicates that the runtime exceeded 24 hours.

5.5.2. Comparison of Algorithms 1, 2 and 3

We now look to see the practical gains from improving the smoother, i.e. the improved smoothing rate of Algorithm 3 should translate into a faster convergence rate [120].

Definition 5.5.2.1. In both Algorithms 2 and 3, we must identify the set $\mathcal{D}$, being pixels at which the coefficients vary significantly. To do this we compute the

minimum multiplicative factor between the largest and smallest of the coefficients $A_{i,j}, B_{i,j}, C_{i,j}, D_{i,j}$ (see §5.4.1). We will denote the minimum multiplicative factor by Σ.

For completion, we will compare Algorithms 2 and 3 to Algorithm 1 for a range of Σ values. The algorithms are all used to segment the image in Figure 5.1(a), with fine grid 1024^2 and coarse grid 32^2 and $\eta = 10^{-4}$ (all parameters are as earlier in §5.5).

Level set energies. In Table 5.7 we give the energy of the level set at the end of each multigrid cycle for the Rada-Chen model for Algorithms 1, 2 and 3 for various Σ values. The rows are ordered in descending order.

	Iteration						
	1	2	3	4	5	6	7
Algorithm 1	2.4687	1.9333	1.9271	1.9253	1.9247	1.9241	1.9236
Algorithm 2 ($\Sigma = 16$)	2.4684	1.9333	1.9264	1.9244	1.9238	-	-
———"——— ($\Sigma = 8$)	2.4683	1.9321	1.9251	1.9242	1.9235	-	-
———"——— ($\Sigma = 4$)	2.4683	1.9302	1.9242	1.9237	1.9226	-	-
———"——— ($\Sigma = 2$)	2.4563	1.9269	1.9214	1.9207	1.9199	-	-
Algorithm 3 ($\Sigma = 16$)	2.4300	1.9185	1.9180	-	-	-	-
———"——— ($\Sigma = 8$)	2.4253	1.9171	1.9166	-	-	-	-
———"——— ($\Sigma = 4$)	2.4184	1.9167	1.9164	-	-	-	-
———"——— ($\Sigma = 2$)	2.4136	1.9165	1.9163	-	-	-	-

Table 5.7: Level set energies ($\times 10^5$) after each multigrid iteration of Algorithms 1, 2 and 3 (for varying Σ) on the image in Figure 5.1(a) + 10% Gaussian noise. A dash indicates convergence before iteration number was reached.

Firstly, we see that Algorithm 3 converges in 3 cycles, where Algorithm 2 converges in 5 and Algorithm 1 converges in 7 cycles. Secondly, we notice that the energy is smallest for Algorithm 3 and Algorithm 2 gives lower energy than Algorithm 1 (for all Σ values). Finally, we notice that as Σ gets smaller (and the number of pixels in $\mathcal{D}$ increases), the energy of the level set at each cycle is smaller. This is all in agreement with the theoretical understanding of the smoothers, that they should give a small rate on the pixels in $\mathcal{D}$, and by increasing the size of $\mathcal{D}$ convergence improves.

Recommended Algorithm. The CPU timings for Algorithm 3 are the best of the three algorithms (Table 5.6). The level set energies are also the lowest for Algorithm 3 (Table 5.7)

at each iteration. It performs the best at tackling the PDEs which have many discontinuous coefficients and the experimental results are in agreement with the theory in §5.4.3. We, therefore, recommend Algorithm 3 to achieve a fast solution to the Rada-Chen and Spencer-Chen selective segmentation models.

Algorithm 3 Results. In Figure 5.7 we briefly show the results of Algorithm 3 applied to the test images for the Rada-Chen model shown in Figure 5.1(a) and Figure 5.5 with $\eta = 10^{-4}$.

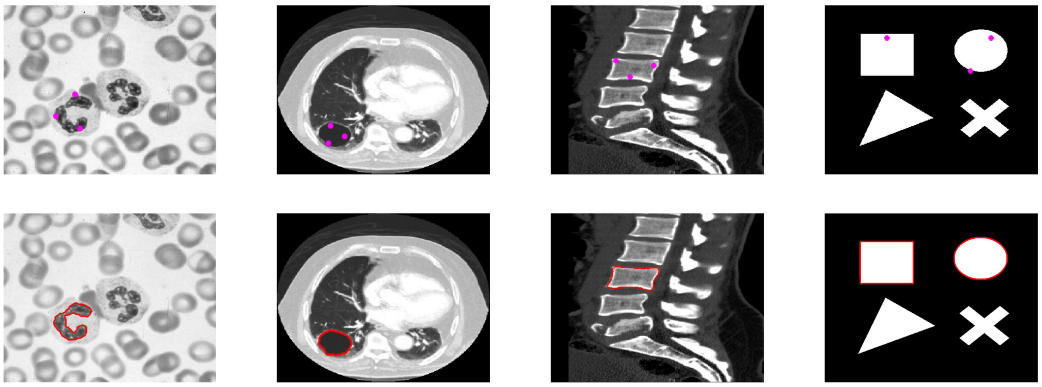


Figure 5.7: Algorithm 3 results; user selections and segmentation results.

5.5.3. Complexity of Algorithm 3

We analyse Algorithm 3 to estimate the complexity of each multigrid cycle. We show analytically and experimentally that Algorithm 3 is $\mathcal{O}(N)$ as is expected for a multigrid method. We start with an analysis of the complexity of the smoother, restriction operator, interpolation operator and coarse grid solver and then use the actual CPU times in Table 5.6 to confirm the predicted complexity.

Analytical complexity. Consider first only the fine grid with $N = nm$ pixels. Hybrid smoother 2 uses GSLEX-I on K pixels and partial line smoothers on L segments, containing the remaining $N - K$ pixels. GSLEX-I requires 13K operations. The partial line smoothers require $\mathcal{O}(M_i)$ operation, where M_i is the size of the line segment for $i \in [0, L]$. Suppose the number of operations for each partial line smoothing is κM_i. We can therefore bound the complexity of the smoothing as $13K + \kappa \sum_{i=0}^{L} M_i$. We know that $K \leq N$

and we perform 4 grid sweeps for every ν_1 pre-smoothing steps and ν_2 post-smoothing steps. For simplicity, assume a square image (i.e. $n = m$) and so for smoothing on one level we have

$$4(\nu_1+\nu_2)\left(13K+\kappa\sum_{i=0}^{L}M_i\right) \leq 4(\nu_1+\nu_2)(13N+\kappa nL) \leq 4(\nu_1+\nu_2)(13+\kappa)N$$

operations. With a V- cycle over T grids, the number of operations is

$$4(\nu_1+\nu_2)(13+\kappa)N(1+\frac{1}{4}+\frac{1}{16}+\cdots+\frac{1}{2^{2(T-1)}}) < 4(13+\kappa)N\frac{\nu_1+\nu_2}{1-2^{-2}} = \frac{16(13+\kappa)(\nu_1+\nu_2)}{3}N$$

The restriction operator has complexity at most $15N$ on the finest grid and with M grids there are $M-1$ restrictions, hence a complexity of less than $20N$. Interpolation has complexity at most $5N$ on the finest grid and hence all interpolation operators contribute at most $\frac{20}{3}N$ operations. Finally, with AOS as the coarse grid solver each iteration needs $448N \cdot 2^{-2(M-1)}$ operations, this is clearly bounded by $448N$. Therefore the overall maximum complexity of Algorithm 3 is

$$\frac{16(13+\kappa)(\nu_1+\nu_2)}{3}N + 20N + \frac{20}{3}N + 448\nu_{AOS}N \leq \left[\frac{16(13+\kappa)(\nu_1+\nu_2)}{3} + 448\nu_{AOS}\right]N,$$

with ν_{AOS} the number of AOS iterations performed - as desired, the algorithm is $\mathcal{O}(N)$.

Experimental complexity. In Table 5.6 we show the ratio of the CPU times for Algorithm 3 on Ω^h when compared with the time on Ω^{2h}. We see that the ratio is around 4 which linearly follows the increase in pixel number. Hence we see experimental confirmation of our analytical result that Algorithm 3 is an $\mathcal{O}(N)$ method.

5.6. Conclusions

Image segmentation models provide a set of challenging and non-linear PDEs with non-smooth coefficients. Direct application of multigrid solvers with standard smoothers such as the lexicographic and line Gauss-Seidel smoothers leads to poor or no convergence. This chapter has investigated the reasons why smoothers become ineffective due to non-smoothness of coefficients and proposed two hybrid smoothers that are aware of jumps and add extra local smoothing using non-standard iterative schemes. We find

that both smoothers lead to convergent multigrid algorithms, however, we recommend one smoother above the other as results are best experimentally and are shown to be good theoretically. Experiments confirm that the proposed new algorithm, outperforms the current fast methods. It also has optimal complexity and therefore is suitable for solving selective segmentation models for large images. Moreover, the ideas used in the design of the new smoother can be applied to other segmentation models and potentially non-smooth PDEs from other applications.

Chapter 6

Conclusions and Future Work

In this thesis, we have presented many new contributions in Chapters 3, 4 and 5. We will now discuss the links between these contributions. In Chapter 3, we considered the distance penalty term used in the variational selective image segmentation framework and proposed to use a modified geodesic distance. This redefines the notion of the "distance" between pixels and objects in an image. This results in a model which is robust to noise, permits anti-markers and can achieve previously unachievable results. It was noted during this work that if the average intensity of the foreground and background are similar, then, although the distance term functions well, it is now the intensity fitting terms which fail and do not perform as desired. This motivated the work of Chapter 4 in which we focussed on improving the intensity terms and proposed a new convex selective segmentation model which incorporates the geodesic distance penalty. We find that this model gives excellent results compared to the competitor models. Not only are the results state-of-the-art, but the model is also very robust to the main parameters, i.e. it achieves good segmentation results for a wide range of parameters. This means that the model is highly applicable and requires minimal parameter tuning from the end user. Finally, in Chapter 5, we lay the groundwork towards a "black-box" non-linear multigrid algorithm for variational selective segmentation models. Although we may have some excellent models, solving the associated equations can be computationally infeasible. Therefore, in this work, we develop a convergent algorithm which is optimal, i.e. computational complexity changes linearly with the number of pixels of the image. This algorithm allows for fast segmentation of large-scale images and with the availabil-

ity and resolution of images only growing larger, this is highly applicable to real-world situations where we require fast segmentation of images in almost real time.

Some extensions to work in this thesis may be to consider

- Using high order regularisers in the models, beyond the TV or weighted-TV regulariser, such as Euler-Elastica which can ensure boundaries which have minimal elastic energy.
- All of our discussed models can be set in a 3D framework too if appropriate for the application. The reformulations are all trivial.
- The models we have proposed are all for greyscale images, it would be possible to generalise these models for segmenting colour images.
- The models we propose can be used to generate training data for use in Deep Learning algorithms for object segmentation. This would avoid manually segmenting each object in the training set.
- The new smoothers proposed in the final chapter can be used to solve more complex non-linear PDEs from other scientific fields where using the non-linear multigrid framework is not currently applied due to non-convergence.
- The models could also be applied to video segmentation or object tracking. Using the segmentation from one frame can form the initialisation for the next frame, allowing the tracking of objects through time.

A highly active area of research within image processing is the use of deep learning (deep neural networks) to solve a range of imaging tasks such as classification, denoising, restoration, deblurring and segmentation. For more information see [24, 74, 106, 147] and the references therein. The use of deep learning for image processing tasks has only been popularly accessible to researchers for the last 5 years, due to the rapid increase in availability of data and the availability of GPUs to researchers. For image segmentation in particular, deep learning is giving excellent results, see [9, 51, 52, 83, 86, 87, 99, 108, 111, 128, 130, 142, 179] (which still represent only a fraction of the recent papers!). Medical image segmentation in the future will be revolutionised by trained deep neural networks, however this does not mean that the traditional image segmentation techniques, discussed in this thesis, are now redundant. To train the deep neural net-

work we require thousands of pre-segmented training images to obtain accurate results. This stage is made significantly faster and easier using these variational models, compared to manual labelling. Recently, the author of this thesis has been working on a project aiming to use deep learning for the segmentation of lungs from CT scans. We used the model developed in Chapter 4 to segment the images semi-automatically. This allowed us to accumulate a large database of segmented lungs very efficiently compared to manual labelling.

With the rapid advances in machine learning, deep learning and artificial intelligence we should, in the near future, be able to alleviate clinicians from repetitive and tedious image labelling, segmentation and quantification tasks. This will not only improve the lives of the clinicians, but will also allow for more improved, rapid and detailed feedback to be obtained from each medical scan. There is also real potential for (i) identifying misdiagnosis, (ii) early disease detection and (iii) detection of additional diseases to those that the patient is being screened for.

Appendix A

Preliminaries Appendix

A.1. Vector Calculus

In this section, we will discuss linear and normed vector spaces and their operators. Vector spaces and their operators will be used and discussed throughout this thesis.

A.1.1. Vector Spaces

Vector spaces are a fundamental algebraic structure. Much of the theory in this thesis relies on spaces which have an underlying vector space structure. We begin by giving the definition of a linear vector space.

Definition A.1.1.1 (Linear Vector Space)**.** Let F be a scalar field (usually of real or complex numbers) and V a set of elements on which two operations, addition and scalar multiplication, have been defined. For $u, v \in V$, the sum of u and v is denoted by $u + v$, and for $c \in F$ a scalar, the scalar multiple of u by c is denoted by cu. If the following ten axioms hold for all $u, v, w \in V$ and for all scalars $c, d \in F$, then V is called a vector space and its elements are called vectors.

1. If $u, v \in V$, then $u + v \in V$ (closure under addition)
2. If $u, v \in V$, then $u + v = v + u$ (commutativity under addition)

3. If $u, v, w \in V$, then $(u + v) + w = u + (v + w)$ (associativity of addition)
4. There exists an element $0 \in V$, called a zero vector, such that $u + 0 = u$ for all $u \in V$ (identity element of addition)
5. For each $u \in V$, there is an element $-u \in V$ such that $u + (-u) = 0$ (existence of additive inverse)
6. If $c \in F$, and $u \in V$, then $cu \in V$ (closure under scalar multiplication)
7. If $u, v \in V$, and $c \in F$ then $c(u + v) = cu + cv$ (distributivity)
8. If $u \in V$, and $c, d \in F$ then $(c + d)u = cu + du$ (distributivity)
9. If $u \in V$ and $c, d \in F$ then $c(du) = (cd)u$ (associativity of scalar multiplication)
10. There exists an element $1 \in V$, called the multiplicative identity, such that $1v = v$ for all $v \in V$ (identity of scalar multiplication)

Example A.1.1.2 (Some vector spaces)**.**

- $\{\mathbf{0}\}$ and $\mathbb{R}^d$ for all $d \in \mathbb{N}$.
- $F[x_1, x_2, \ldots, x_d]$, i.e. the set of polynomials with variables $x_1, x_2, \ldots, x_d$ with coefficients in F.
- The set of differentiable functions.
- $BV(\Omega)$.

All vector spaces we consider in this thesis are linear spaces. We will refer to "linear vector spaces" as "vector spaces" throughout.

A.1.2. Normed Linear Spaces

We extend the idea of a vector space by equipping them with a *norm*. This gives a notion of distance between elements in the vector space.

Definition A.1.2.1 (Norm)**.** For a given a vector space V over a subfield $F \subseteq \mathbb{C}$, a real-valued function $N : V \to \mathbb{R}$ is called a norm on V if for all $a \in F$ and all $u, v \in V$, it satisfies

1. $N(\alpha v) = |\alpha| N(v)$ for all $\alpha \in \mathbb{R}$ and $v \in V$.
2. $N(v + u) \leq N(v) + N(u)$ for all $v, u \in V$.
3. $N(v) \geq 0$ for all $v \in V$.
4. $N(v) = 0$ if $v = 0$.

A norm is a seminorm if the fourth property is neglected. A norm on a vector space V induces a metric on V by

$$d(v, u) := N(v - u).$$

This metric is invariant under translations and is absolutely homogeneous, i.e. $d(v + w, u + w) = d(v, u)$ and $d(\lambda v, \lambda u) = |\lambda| d(v, u)$. The norm of a vector $\boldsymbol{x} \in \mathbb{R}^d$ is usually represented by $||\boldsymbol{x}||$.

Example A.1.2.2 (Examples of norms)**.**

- The absolute value is a norm on the set of real numbers $\mathbb{R}$.
- Euclidean norm of a vector: Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ then

  $$||\mathbf{x}|| = \sqrt{x_1^2 + x_2^2 + \ldots + x_d^2}.$$

  This gives the ordinary distance from the origin to the point $\boldsymbol{x}$.
- Infinity norm ℓ_∞: For $\boldsymbol{x} \in \mathbb{R}^d$, $||\boldsymbol{x}||_\infty = \max\{|x_1|, |x_2|, \ldots, |x_d|\}$.
- ℓ_p-norm of a vector: Consider $\boldsymbol{x} \in \mathbb{R}^d$, then for any real number $p \geq 1$ the ℓ_p-norm of $\boldsymbol{x}$ is defined as

  $$||\boldsymbol{x}||_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p},$$

  Clearly, for $p = 2$ this is the Euclidean norm, and as $p \to \infty$ we see $|| \cdot ||_p \to$

$||\cdot||_\infty$.

- L^p-norm of a function: Consider a continuous function f defined on a domain Ω such that $\int_\Omega |f(x)|^p \, \mathrm{d}x < \infty$ with $1 \leq p \leq \infty$. Then

$$||f(x)||_{L^p} = \left(\int_\Omega |f(x)|^p \, \mathrm{d}x \right)^{1/p}$$

  defines the L^p-norm of f on Ω. The special case when $p = \infty$ is defined as $||f(x)||_{L^\infty} = \sup_x |f(x)|$.
- The Total Variation (TV) of $u : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$, given by (2.1), defines a norm.

Definition A.1.2.3 (Normed Linear Space)**.** A vector space equipped with a norm (seminorm) $||\cdot||$ defined on it is called a normed linear space (seminormed linear space).

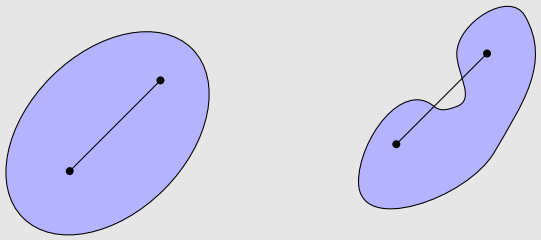
A.1.3. Convex Sets and Functions

We will now discuss the notions of a convex set and a convex function. Informally, for a convex set, any two elements in that set can be joined by a line which lies entirely within the set. For a convex function, we naturally obtain the result that it will have a unique maximum or minimum.

Definition A.1.3.1 (Convex Set)**.** A set $\mathcal{S}$ in a vector space V is said to be convex if, for all $u, v \in \mathcal{S}$ and all $\theta \in [0,1]$, the point $w = (1-\theta)u + \theta v \in \mathcal{S}$. In other words, every point on the line segment connecting u and v is in $\mathcal{S}$.

Example A.1.3.2 (Convex and Non-Convex Sets)**.** In Figure A.1, we give examples of (a) convex and (b) non-convex sets. We see for the convex set that any line segment is within the set, whereas for the non-convex set we can find points on the line which

lie outside the set.



(a) Convex Set (b) Non-Convex Set

Figure A.1: Example convex and non-convex sets.

Definition A.1.3.3 (Convex Functions)**.** A function $f : \mathcal{S} \to \mathbb{R}$ defined on a convex set $\mathcal{S}$ of some vector space is called convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) \tag{A.1}$$

for all $x, y \in \mathcal{S}$ and $\theta \in (0,1)$. If the inequality is always strict for $x \neq y$, f is called strictly convex.

Example A.1.3.4 (Proving convexity)**.** Let us consider the function $f(x) = x^2$. For $x, y \in \mathbb{R}$ and $\theta \in [0,1]$ we have

$$\begin{aligned}(\theta f(x) + (1-\theta)f(y)) - (f(\theta x + (1-\theta)y)) &= \left(\theta x^2 + (1-\theta)y^2\right) - (\theta x + (1-\theta)\, y)^2 \\ &= \theta(1-\theta)x^2 + (1-\theta)\theta y^2 + 2\theta(1-\theta)xy \\ &= \theta(1-\theta)\left(x^2 + y^2 + 2xy\right) \\ &= \theta(1-\theta)\,(x+y)^2 \geq 0\end{aligned}$$

and therefore $f(x)$ is convex. We give the plot of $f(x)$ and show how more intuitively how the inequality holds for $x = -0.5$ and $y = 1$.

Figure A.2: Plot of $f(x) = x^2$

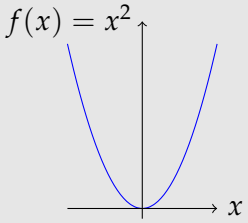
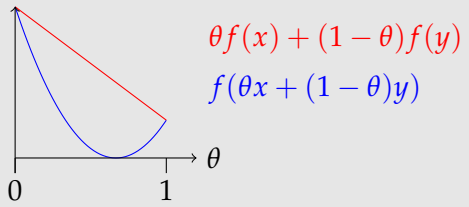


Figure A.3: A graph showing (A.1) holds for $x = -0.5$ and $y = 1$.



Example A.1.3.5 (Examples of Convex Functions on $\mathbb{R}$ and $\mathbb{R}^d$)**.**

- The exponential function $f(x) = \exp(ax) = e^{ax}$, for any $a \in \mathbb{R}$ on domain $\mathbb{R}$ is convex.
- The norms $||\boldsymbol{x}||_p = \left(\sum_{i=1}^n |x_i|_p\right)^{1/p}$, for $p \geq 1$, and $||\boldsymbol{x}||_\infty = \max_k(|x_k|)$ are convex.

Definition A.1.3.6 (Functional)**.** A functional $\mathcal{F}$ is a mapping from a space X into scalar field F.

Remark A.1.3.7. We will only consider functionals mapping from a vector space X into the field $F = \mathbb{R}$. The notion of convexity from Definition A.1.3.3 also applies to functionals, i.e. if functional $\mathcal{F}$ satisfies

$$\mathcal{F}(\theta u + (1-\theta)v) \leq \theta\mathcal{F}(u) + (1-\theta)\mathcal{F}(v)$$

for all $u, v \in X$ we say that $\mathcal{F}$ is a convex functional.

Example A.1.3.8 (Examples of Convex Functionals)**.**

- The TV norm of $u : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ given by (2.1) is a convex functional.
- For $u, v \in \mathbb{R}^d$ with v fixed, the functional

$$\mathcal{F}(u) = ||u - v||_{L^2}^2$$

is convex.

Example A.1.3.9 (Proving TV is convex)**.** Let us define $\mathcal{F}(u)$ for $u \in BV(\Omega)$ as the TV functional, i.e.

$$\mathcal{F}(u) := TV(u;\Omega) = \int_\Omega |\nabla u| \, \mathrm{d}\Omega, \tag{A.2}$$

and we have

$$\begin{aligned} \mathcal{F}(\theta u + (1-\theta)v) &= \int_\Omega |\nabla \left(\theta u + (1-\theta)v\right)| \, \mathrm{d}\Omega, \\ &= \int_\Omega |\theta \nabla u + (1-\theta)\nabla v| \, \mathrm{d}\Omega, \\ &\leq \int_\Omega \left(\theta |\nabla u| + (1-\theta)|\nabla v|\right) \mathrm{d}\Omega, \\ &= \theta \mathcal{F}(u) + (1-\theta)\mathcal{F}(v), \end{aligned} \tag{A.3}$$

and therefore $TV(u;\Omega)$ is a convex functional.

A.1.4. Vector Operators

Here, we will give some definitions of the vector operators which we use throughout this thesis.

Definition A.1.4.1. For a given scalar function $\phi : \mathbb{R}^d \to \mathbb{R}$ the gradient, denoted by $\nabla\phi$, is defined as

$$\nabla\phi = \left(\frac{\partial\phi}{\partial x_1}, \frac{\partial\phi}{\partial x_2}, \ldots, \frac{\partial\phi}{\partial x_d}\right)$$

The gradient $\nabla\phi$ is perpendicular to the tangents of ϕ and points in the direction of maximum increase in ϕ. The unit (outward) normal vector $\boldsymbol{n}$ is a vector that points in the same direction as the gradient $\nabla\phi$ for points on the surface of ϕ and is defined as

$$\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}$$

Definition A.1.4.2. The divergence of $\phi : \mathbb{R}^d \to \mathbb{R}$, denoted by $\nabla \cdot \phi$, is defined as

$$\nabla \cdot \phi = \frac{\partial \phi}{\partial x_1} + \frac{\partial \phi}{\partial x_2} + \ldots + \frac{\partial \phi}{\partial x_d}$$

Definition A.1.4.3. The mean curvature of $\phi : \mathbb{R}^d \to \mathbb{R}$ is defined as the divergence of the unit normal $\boldsymbol{n}$

$$\kappa = \nabla \cdot \boldsymbol{n} = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} = \frac{\partial}{\partial x_1}\left(\frac{1}{|\nabla \phi|}\frac{\partial \phi}{\partial x_1}\right) + \frac{\partial}{\partial x_2}\left(\frac{1}{|\nabla \phi|}\frac{\partial \phi}{\partial x_2}\right) + \ldots + \frac{\partial}{\partial x_d}\left(\frac{1}{|\nabla \phi|}\frac{\partial \phi}{\partial x_d}\right)$$

Definition A.1.4.4 (Laplacian)**.** The Laplacian of $\phi : \mathbb{R}^d \to \mathbb{R}$, denoted by $\Delta\phi$, is defined as

$$\Delta \phi = \sum_{i=1}^{d} \frac{\partial^2 \phi}{\partial x_i^2}$$

In this thesis we only consider $d = 2$ and $d = 3$ for which $\Delta\phi = \phi_{xx} + \phi_{yy}$ and $\Delta\phi = \phi_{xx} + \phi_{yy} + \phi_{zz}$ respectively.

A.1.5. Divergence Theorem

The divergence theorem (also known as Gauss' Theorem or the Gauss-Ostrogradsky Theorem) is a fundamental result in vector calculus. This theorem relates the integral of the divergence of the vector field over domain Ω to a surface integral over the domain boundary $\partial\Omega$. This is extensively applied in Calculus of Variations and will be revisited in §A.2.2.

Theorem A.1.5.1 (The Divergence Theorem)**.** Let F be a continuously differentiable vector function in a domain $V \subset \mathbb{R}^d$. Let $\Omega \subset V$ be a closed, bounded region whose boundary is a smooth surface $\partial\Omega$. Then

$$\int_\Omega (\nabla \cdot F)\, \mathrm{d}\Omega = \int_{\partial\Omega} F \cdot \boldsymbol{n}\, \mathrm{d}s$$

where ds indicates integration with respect to surface area on $\partial\Omega$, and $\boldsymbol{n}$ is the unit

outward normal vector for each point $\boldsymbol{x} \in \partial\Omega$.

A.1.6. Coarea Formula

The coarea formula gives a natural connection between the total variation of a function $u(\boldsymbol{x})$ and the length of its level sets. For an open set in Euclidean space, the coarea formula states that the total variation of a function can be computed by integrating the lengths of its level sets. For a function $u(\boldsymbol{x}) \in BV(\Omega)$ defined in open $\Omega \subset \mathbb{R}^d$, we define a cumulative level set E_λ by

$$E_\lambda = \{\boldsymbol{x} \in \Omega : u(\boldsymbol{x}) \geq \lambda\}.$$

Definition A.1.6.1 (Perimeter)**.** The perimeter of $E_\lambda \in \Omega$ is defined

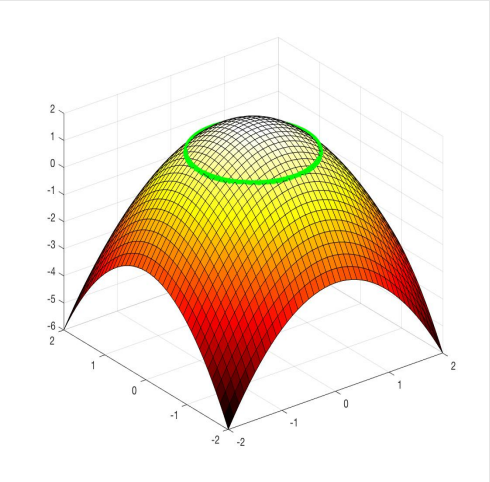
$$\text{Per}(E_\lambda) = \int_{\partial\chi_{E_\lambda}} \text{d}s$$

where χ_{E_λ} is the characteristic function of the set E_λ defined

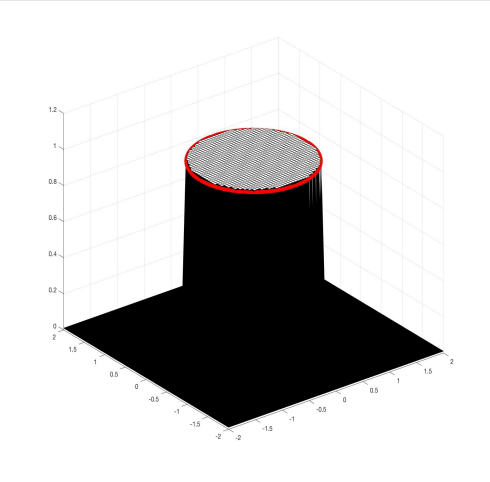
$$\chi_{E_\lambda} = \begin{cases} 1, & \text{for } \boldsymbol{x} \in E_\lambda \\ 0, & \text{for } \boldsymbol{x} \notin E_\lambda \end{cases}$$

Example A.1.6.2 (E_λ and χ_{E_λ} for a level set)**.** We see from the figure below that the

length of $\partial\chi_{E_\lambda}$ and the length of the level set $\{u(\boldsymbol{x}) = \lambda\}$ are equal.



(a) Arbitrary function $u(\boldsymbol{x})$ with the level set $\{u(\boldsymbol{x}) = 1\}$ indicated in green.



(b) The corresponding plot of χ_{E_λ} for $\lambda = 1$ and $\partial\chi_{E_\lambda}$ indicated in red.

Figure A.4: We give a more intuitive understanding of χ_{E_λ} and $\partial\chi_{E_\lambda}$ for an arbitrary function.

Definition A.1.6.3 (Lipschitz Continuity)**.** A function $u : \Omega \subset \mathbb{R}^n \to \mathbb{R}^n$, on open Ω, is Lipschitz continuous on Ω if there exists a positive constant $\mathcal{K} \in \mathbb{R}_{>0}$ such that for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$

$$||u(\boldsymbol{x}) - u(\boldsymbol{y})||_* \leq \mathcal{K}||\boldsymbol{x} - \boldsymbol{y}||_*.$$

The constant $\mathcal{K}$ is called the Lipschitz constant and the norm $||\cdot||_*$ is arbitrary.

Theorem A.1.6.4 (Coarea Formula)**.** Let $\Omega \subset \mathbb{R}^d$ be an open set and $u(\boldsymbol{x}) : \Omega \to \mathbb{R}$ be Lipschitz continuous, then for $g(\boldsymbol{x}) \in L^1(\Omega)$

$$\int_\Omega g(\boldsymbol{x})|\nabla u(\boldsymbol{x})| \, \mathrm{d}x = \int_{-\infty}^{+\infty} \left(\int_{\partial\chi_{E_\lambda}} g(\boldsymbol{x}) \, \mathrm{d}s \right) \mathrm{d}\lambda$$

For the particular case when $g(\boldsymbol{x}) = 1$ we have

$$\int_{\Omega} |\nabla u(\boldsymbol{x})| \mathrm{d}x = \int_{-\infty}^{+\infty} \left(\int_{\partial \chi_{E_\lambda}} \mathrm{d}s \right) \mathrm{d}\lambda = \int_{-\infty}^{+\infty} \mathrm{Per}(E_\lambda) \, \mathrm{d}\lambda.$$

The proof for the coarea formula is complicated and contained in [7, 63, 65, 182]. In the context of imaging, using this formula, the total variation of $u(\boldsymbol{x}) \in BV(\Omega)$ is the sum of the lengths of all level sets. This means that discontinuities in $u(\boldsymbol{x})$ are accounted for.

A.2. Calculus of Variations

Many imaging problems require finding the minimiser of some functional $\mathcal{F}(u)$. The functional is typically an integral for which the minimiser u has desirable properties (e.g. smooth, piecewise constant, piecewise smooth). Calculus of Variations allows us to find the minimiser for a given functional and is fundamental in many branches of mathematics, physics, engineering and other fields. Extensive literature can be found discussing Calculus of Variations and its applications; we refer the reader to [69, 70, 71, 146] for a good introduction. In this section, we introduce the basic tools to compute the first variation of a functional, also known as the Euler-Lagrange equation. The solution to the Euler-Lagrange equation gives us the minimiser for the given functional.

A.2.1. Gâteaux Derivative

The Gâteaux Derivative is a generalisation of the idea of the directional derivative.

Definition A.2.1.1 (Gâteaux derivative)**.** Let U and V be vector spaces and $\mathcal{F} : U \to V$ be a function/functional. Let $\boldsymbol{u}, \boldsymbol{\psi} \in U$ be vectors with $\boldsymbol{\psi} \neq \boldsymbol{0}$. The Gâteaux derivative of $\mathcal{F}$ is

$$\delta \mathcal{F}(\boldsymbol{u}; \boldsymbol{\psi}) = \lim_{\varepsilon \to 0} \frac{\mathcal{F}(\boldsymbol{u} + \varepsilon \boldsymbol{\psi}) - \mathcal{F}(\boldsymbol{u})}{\varepsilon}$$

and if $\delta \mathcal{F}(\boldsymbol{u}; \boldsymbol{\psi})$ exists for $\boldsymbol{u}$, we say that $\mathcal{F}(\boldsymbol{u})$ is Gâteaux differentiable at $\boldsymbol{u}$.

Example A.2.1.2 (The exponential function)**.** Let us consider the exponential function $f(\boldsymbol{u}) = e^{\boldsymbol{u}} : \mathbb{R} \to \mathbb{R}^+$ which can be written as the infinite series

$$\exp(\boldsymbol{u}) = \sum_{k=0}^{\infty} \frac{\boldsymbol{u}^k}{k!} = 1 + \boldsymbol{u} + \frac{\boldsymbol{u}^2}{2!} + \cdots$$

By straightforward calculation, given $\exp(a+b) = \exp(a)\exp(b)$ and the infinite series expansion, we can compute the Gâteaux derivative

$$\delta \exp(\boldsymbol{u}; \boldsymbol{\psi}) = \lim_{\varepsilon \to 0} \frac{e^{\boldsymbol{u}} e^{\varepsilon \boldsymbol{\psi}} - e^{\boldsymbol{u}}}{\varepsilon} = e^{\boldsymbol{u}} \lim_{\varepsilon \to 0} \frac{e^{\varepsilon \boldsymbol{\psi}} - 1}{\varepsilon} = e^{\boldsymbol{u}} \lim_{\varepsilon \to 0} \left[\boldsymbol{\psi} + \frac{\boldsymbol{\psi}^2}{2} \varepsilon + \mathcal{O}(\varepsilon^2) \right] = e^{\boldsymbol{u}} \boldsymbol{\psi}$$

In the following example, we obtain the Gâteaux Derivative for one of the most common functionals in imaging, the *total variation*.

Example A.2.1.3 (Gâteaux Derivative for the Total Variation functional)**.** The TV functional is

$$TV(\boldsymbol{u}) = \int_{\Omega} |\nabla \boldsymbol{u}(\boldsymbol{x})| \, \mathrm{d}\Omega,$$

and the Gâteaux derivative is determined as follows

$$\begin{aligned}
\delta TV(\boldsymbol{u}; \boldsymbol{\psi}) &= \lim_{\varepsilon \to 0} \frac{TV(\boldsymbol{u} + \varepsilon \boldsymbol{\psi}) - TV(\boldsymbol{u})}{\varepsilon} = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \int_{\Omega} \left(|\nabla \left(\boldsymbol{u} + \varepsilon \boldsymbol{\psi} \right)| - |\nabla \boldsymbol{u}| \right) \mathrm{d}\Omega \\
&= \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \int_{\Omega} \left(|\nabla \boldsymbol{u}| + \varepsilon \frac{\nabla \boldsymbol{u} \cdot \nabla \boldsymbol{\psi}}{|\nabla \boldsymbol{u}|} + \mathcal{O}(\varepsilon^2) - |\nabla \boldsymbol{u}| \right) \mathrm{d}\Omega \\
&= \lim_{\varepsilon \to 0} \int_{\Omega} \left(\frac{\nabla \boldsymbol{u} \cdot \nabla \boldsymbol{\psi}}{|\nabla \boldsymbol{u}|} + \mathcal{O}(\varepsilon) \right) \mathrm{d}\Omega = \int_{\Omega} \frac{\nabla \boldsymbol{u} \cdot \nabla \boldsymbol{\psi}}{|\nabla \boldsymbol{u}|} \, \mathrm{d}\Omega
\end{aligned} \tag{A.4}$$

and we use Green's Identity

$$\int_{\partial \Omega} \frac{\nabla \boldsymbol{u} \cdot \boldsymbol{n}}{|\nabla \boldsymbol{u}|} \boldsymbol{\psi} \, \mathrm{d}s = \int_{\Omega} \frac{\nabla \boldsymbol{u} \cdot \nabla \boldsymbol{\psi}}{|\nabla \boldsymbol{u}|} \, \mathrm{d}\Omega + \int_{\Omega} \nabla \cdot \frac{\nabla \boldsymbol{u}}{|\nabla \boldsymbol{u}|} \boldsymbol{\psi} \, \mathrm{d}\Omega$$

Therefore

$$\delta TV(\boldsymbol{u};\boldsymbol{\psi}) = \int_{\partial\Omega} \frac{\nabla \boldsymbol{u}\cdot\boldsymbol{n}}{|\nabla \boldsymbol{u}|}\boldsymbol{\psi}\ \mathrm{d}s - \int_{\Omega} \nabla\cdot\frac{\nabla \boldsymbol{u}}{|\nabla \boldsymbol{u}|}\boldsymbol{\psi}\ \mathrm{d}\Omega. \tag{A.5}$$

Definition A.2.1.4 (Local Minimiser)**.** A real-valued functional $\mathcal{F} : U \to \mathbb{R}$, defined on a subset U of the normed space V, is said to have a local minimiser at the point $\tilde{u} \in U$, relative to the norm $||\cdot||$, if there exists some $\varepsilon > 0$ such that

$$\mathcal{F}(\tilde{u}) \leq \mathcal{F}(u), \qquad \forall u \in B_\varepsilon(\tilde{u}) \cap U, \tag{A.6}$$

with $B_\varepsilon(\tilde{u}) := \{u \in U : ||u - \tilde{u}|| < \varepsilon\}$ a ball of radius ε around $\tilde{u}$.

In the same way, the local maximiser can be defined by replacing the inequality (A.6) with $\mathcal{F}(\tilde{u}) \geq \mathcal{F}(u)$.

Definition A.2.1.5 (Global Minimiser)**.** A real-valued functional $\mathcal{F} : U \to \mathbb{R}$ is said to have a global minimiser at the point $\tilde{u} \in U$ if $\mathcal{F}(\tilde{u}) \leq \mathcal{F}(u)$ for all $u \in U$.

Similarly, we obtain the global maximiser if we have the inequality $\mathcal{F}(\tilde{u}) \geq \mathcal{F}(u)$.

Definition A.2.1.6 (Stationary Point)**.** Let $\mathcal{F} : U \to \mathbb{R}$ be a function with solution space $U \subset V$. Suppose that for some $\tilde{u} \in U$, $\mathcal{F}$ is Gâteaux differentiable for all arbitrary functions $\boldsymbol{\psi} \in V$. Then $\tilde{u} \in U$ is said to be a stationary point of $\mathcal{F}$ if $\delta\mathcal{F}(\tilde{u};\boldsymbol{\psi}) = 0$ for all $\boldsymbol{\psi} \in V$.

A.2.2. Euler-Lagrange Equation

In this section, we will introduce a critical concept within Calculus of Variations, namely the Euler-Lagrange equation. This is the equation whose solution is a minimiser for its corresponding functional. In variational image segmentation, the Euler-Lagrange equation is typically a PDE which we must solve, whose solution gives us the segmentation boundary. This will be discussed later in §A.3.

Definition A.2.2.1 (Euler-Lagrange Equation)**.** The equation $\delta\mathcal{F}(u,\psi) = 0$ is called the Euler-Lagrange equation of the minimisation problem $\min_{u\in U} \mathcal{F}(u)$.

Theorem A.2.2.2 (Necessary Condition for a Local Minimiser)**.** For a given Gâteaux differentiable function $\mathcal{F} : U \to \mathbb{R}$, if $\tilde{u}$ is a local minimiser of $\mathcal{F}(u)$, then $\tilde{u}$ is a stationary point of $\mathcal{F}(u)$.

The proof of this theorem can be found in [58].

Example A.2.2.3 (Euler-Lagrange equation for L^2 difference)**.** Let $\mathcal{F} : U \to \mathbb{R}$ and $\boldsymbol{u}, \boldsymbol{z} \in \mathcal{U}$, define

$$\begin{aligned}\mathcal{F}(\boldsymbol{u}) &= ||\boldsymbol{u} - \boldsymbol{z}||_2^2 \\ \delta\mathcal{F}(\boldsymbol{u};\boldsymbol{\psi}) &= \lim_{\varepsilon\to 0} \frac{1}{\varepsilon}\left(||\boldsymbol{u} + \varepsilon\boldsymbol{\psi} - \boldsymbol{z}||_2^2 - ||\boldsymbol{u} - \boldsymbol{z}||_2^2\right) \\ &= \lim_{\varepsilon\to 0} \frac{1}{\varepsilon}\left(2\varepsilon\langle \boldsymbol{u} - \boldsymbol{z}, \boldsymbol{\psi}\rangle_2 + \varepsilon^2\boldsymbol{\psi}\right) \\ &= 2\langle \boldsymbol{u} - \boldsymbol{z}, \boldsymbol{\psi}\rangle_2\end{aligned}$$

We set $\delta\mathcal{F}(\boldsymbol{u};\boldsymbol{\psi}) = 0$ and $\boldsymbol{\psi}$ is arbitrary, therefore the Euler-Lagrange equation is

$$\boldsymbol{u} - \boldsymbol{z} = 0.$$

i.e. at the minimum of $\mathcal{F}(\boldsymbol{u})$, we have $\boldsymbol{u} = \boldsymbol{z}$.

Example A.2.2.4 (Euler-Lagrange Equation for TV)**.** Consider again the TV functional (2.1). As we found earlier in Example A.2.1.3, the Gâteaux derivative is

$$\delta TV(\boldsymbol{u};\boldsymbol{\psi}) = \int_{\partial\Omega} \frac{\nabla\boldsymbol{u}\cdot\boldsymbol{n}}{|\nabla\boldsymbol{u}|}\boldsymbol{\psi}\ \mathrm{d}s - \int_\Omega \nabla\cdot\frac{\nabla\boldsymbol{u}}{|\nabla\boldsymbol{u}|}\boldsymbol{\psi}\ \mathrm{d}\Omega \tag{A.7}$$

and if we suppose Neumann boundary conditions, i.e. $\nabla\boldsymbol{u}\cdot\boldsymbol{n} = \frac{\partial\boldsymbol{u}}{\partial\boldsymbol{n}} = \boldsymbol{0}$, then we obtain

$$\delta TV(\boldsymbol{u};\boldsymbol{\psi}) = -\int_\Omega \nabla\cdot\frac{\nabla\boldsymbol{u}}{|\nabla\boldsymbol{u}|}\boldsymbol{\psi}\ \mathrm{d}\Omega.$$

Therefore, as $\boldsymbol{\psi}$ is arbitrary and $\boldsymbol{\psi} \neq \mathbf{0}$, the solution to $\delta TV(\boldsymbol{u}; \boldsymbol{\psi}) = 0$ is

$$\nabla \cdot \frac{\nabla \boldsymbol{u}}{|\nabla \boldsymbol{u}|} = \mathbf{0}.$$

A.3. Discretised Framework

In general, a continuous linear boundary value problem (a linear partial differential equation with boundary conditions specified) in d-dimensions is denoted by

$$\mathcal{L}_\Omega u(\boldsymbol{x}) = f_\Omega(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \Omega, \qquad \mathcal{L}_{\partial\Omega} u(\boldsymbol{x}) = f_{\partial\Omega}(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \partial\Omega,$$

where $\mathcal{L}_\Omega$ and $\mathcal{L}_{\partial\Omega}$ are a linear operators, Ω is a bounded open domain in $\mathbb{R}^d, \boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ and $\partial\Omega$ is the boundary of Ω. For given Ω, $\mathcal{L}_\Omega$, $\mathcal{L}_{\partial\Omega}$, f_Ω and $f_{\partial\Omega}$, we aim to find the value of u for all $\boldsymbol{x}$.

Example A.3.0.1. One of the most simple examples of a linear boundary problem is Poisson's equation with the boundary values of $u(\boldsymbol{x})$ determined by the imposed boundary conditions

$$-\Delta_\Omega u(\boldsymbol{x}) = f_\Omega(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \Omega, \qquad u(\boldsymbol{x}) = f_{\partial\Omega}(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \partial\Omega, \tag{A.8}$$

Similarly, a continuous non-linear boundary value problem is defined by

$$\mathcal{N}_\Omega u(\boldsymbol{x}) = f_\Omega(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \Omega, \qquad \mathcal{N}_{\partial\Omega} u(\boldsymbol{x}) = f_{\partial\Omega}(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \partial\Omega,$$

where $\mathcal{N}_\Omega$ and $\mathcal{N}_{\partial\Omega}$ are non-linear operators and Ω and $\partial\Omega$ are as before. Again, for given Ω, $\mathcal{N}_\Omega$, $\mathcal{N}_{\partial\Omega}$, f_Ω and $f_{\partial\Omega}$ we want to determine the value of u for all $\boldsymbol{x}$.

However, it is not so simple. Often when solving a partial differential equation (PDE) it will not be possible to obtain a solution analytically. The Euler-Lagrange equations which result from the imaging models, such as (2.12), (2.17) and (2.19), tend to have no analytical solution. Therefore we seek to solve only a discretised version of the continuous PDE, i.e. we solve the PDE for a discretised Ω.

In this thesis, we deal with image domains which are usually square or rectangular and

where the values of $f(\boldsymbol{x})$ are known for uniformly distributed points $\boldsymbol{x} \in \Omega \subset \mathbb{R}^d$. Therefore, a natural choice for discretising the domain is to use the finite difference method. We will restrict our discussion to $\Omega = [0,1]^d \subset \mathbb{R}^d$ in this thesis. The discussion will also focus on two-dimensional Ω, i.e. $d = 2$, however, the theory readily extends to higher dimensions.

Two-dimensional discretisation. To discretise Ω in two dimensions, we must divide $\Omega = [0,1] \times [0,1]$ into grid points. We may discretise Ω in many ways, two popular choices, which we will discuss, are the vertex-centered and cell-centered discretisations.

In both cases, we choose positive integers n and m and define step lengths in the x and y directions as $h_x = 1/n$ and $h_y = 1/m$ respectively.

Vertex-Centered Grid. The grid is discretised as

$$G_{n,m} = \left\{ \left(ih_x, jh_y\right) \in [0,1]^2 \;\middle|\; 0 \leq i \leq n, 0 \leq j \leq m \right\},$$

which has dimensions $(n+1) \times (m+1)$. See Figure A.5(a).

Cell-Centered Grid. The grid is discretised as follows

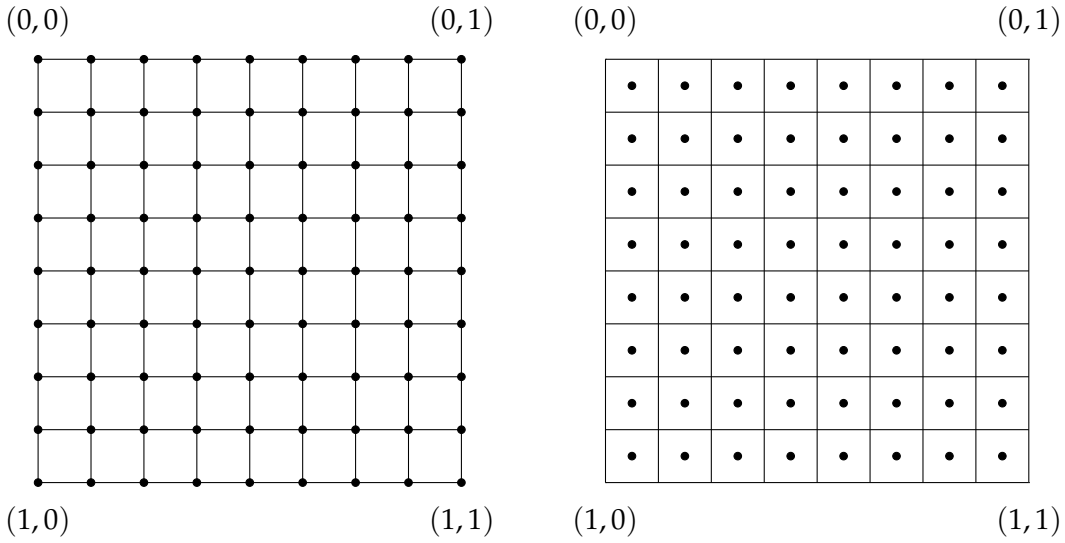
$$G_{n,m} = \left\{ \left((i-1/2)h_x, (j-1/2)h_y\right) \in [0,1]^2 \;\middle|\; 1 \leq i \leq n, 1 \leq j \leq m \right\},$$

which has dimensions $n \times m$. See Figure A.5(b).

These definitions both easily extend into higher dimensions. In this thesis, we will only discuss the cell-centered approach. This is due to the fact that images are given in dimensions $n \times m$ (2D) or $n \times m \times k$ (3D) and these dimensions are preserved using the cell-centered approach. However, the vertex-centered approach is perfectly justifiable to use without consequence and the choice is personal preference.

Boundary Conditions (BCs). The boundary points $\partial\Omega$ are defined as the set of mesh points in $\mathbb{R}^d$ which don't belong to Ω, but which have a nearest neighbour in Ω. There are two types of boundary conditions which we will discuss; Dirichlet and Neumann.

Dirichlet BCs. In this case, we impose the values which the solution must take at the boundary. Commonly, Dirichlet boundary conditions impose a fixed constant value or a function value at those points.



(a) Vertex-Centered Discretisation (b) Cell-Centered Discretisation

Figure A.5: Two common discretisations for $\Omega = [0,1]^2$. We indicate the coordinates of the outer corners of the domain. The dots indicate points of $G_{n,m}$ in the different discretisations.

Neumann BCs. In this case, we impose the value that the derivative of the solution must take at the boundary points. It is common for imaging PDEs to impose zero Neumann BCs, i.e. $\frac{\partial u}{\partial \boldsymbol{n}} = 0$ where $\boldsymbol{n}$ is the outward unit normal to $\partial\Omega$.

A.3.1. Finite Differences

We will now discuss the finite difference technique we use to obtain approximations to the derivatives on a discretised grid.

Remark A.3.1.1. From this point onwards, for notational convenience, we will typically only consider square grids in 2D, i.e. those for which $n = m$ and $h_x = h_y =: h$. However, we remark that the finite difference equations also hold for rectangular images and that the equations for finite differences in 3D are similarly derived.

Notation A.3.1.2. At this point, it is useful to make some notation conventions explicit:

- For a cell-centered domain discretised by step length h in each dimension, the interior of the discrete grid is denoted by Ω^h and the boundary by $\partial\Omega^h$.
- We denote by (x_i, y_j) the grid point corresponding to $((i-1/2)h, (j-1/2)h) \in \Omega^h$.
- We denote by $u_{i,j}$ the value of $u(x_i, y_j)$.
- To be explicit, the bold $\boldsymbol{x} \in \mathbb{R}^2$ refers to a coordinate in Ω and $x \in \mathbb{R}$ refers to a number.

With the cell-centered grid in place, the operators in the PDE can be approximated locally using the Taylor series expansion, i.e. from

$$u(x_i + h, y_j) = u(x_i, y_j) + h\frac{\partial u}{\partial x}\Big|_{(x_i,y_j)} + \mathcal{O}(h^2) \tag{A.9}$$

we can approximate the operator $\frac{\partial u}{\partial x}$ at the grid point $(i, j) \in \mathbb{N}^2$ by

$$\frac{\partial u}{\partial x}\Big|_{(x_i,y_j)} = \frac{u(x_i + h, y_j) - u(x_i, y_j)}{h} + \mathcal{O}(h).$$

We can assume that h is small and therefore assume all terms in $\mathcal{O}(h)$ are small and can be neglected, therefore we obtain the first-order forward difference operator

$$\Delta_x^+ u_{i,j} \approx \frac{u(x_i + h, y_j) - u(x_i, y_j)}{h} = \frac{u_{i+1,j} - u_{i,j}}{h}.$$

Similarly, by using the Taylor expansion of $u(x_i - h, y_j)$, i.e.

$$u(x_i - h, y_j) = u(x_i, y_j) - h\frac{\partial u}{\partial x}\Big|_{(x_i,y_j)} + \mathcal{O}(h^2) \tag{A.10}$$

we obtain the first-order backward difference operator

$$\Delta_x^- u_{i,j} \approx \frac{u(x_i, y_j) - u(x_i - h, y_j)}{h} = \frac{u_{i,j} - u_{i-1,j}}{h}$$

By subtracting the Taylor expansion (A.10) from (A.9) we obtain

$$u(x_i + h, y_j) - u(x_i - h, y_j) = 2h \frac{\partial u}{\partial x}\Big|_{(x_i, y_j)} + \mathcal{O}(h^3)$$

and from this, we get the second-order central difference approximation

$$\Delta_x^c u_{i,j} \approx \frac{u(x_i + h, y_j) - u(x_i - h, y_j)}{2h} = \frac{u_{i+1,j} - u_{i-1,j}}{2h}$$

Similarly, using the Taylor expansion again, we can obtain the approximations of higher-order derivatives. For example, a second-order approximation to $\frac{\partial^2 u}{\partial x^2}$ at (i, j) is given by

$$u_{xx}\Big|_{(x_i, y_j)} \approx \frac{u(x_i + h, y_j) - 2u(x_i, y_j) + u(x_i - h, y_j)}{h^2}$$

and $\frac{\partial^2 u}{\partial y^2}$ can be defined in a similar way.

Notation A.3.1.3. Using the finite differences to approximate $\mathcal{L}u$, we obtain the discrete approximation to the continuous problem. The discrete problem is denoted

$$\mathcal{L}_\Omega^h u^h(\boldsymbol{x}) = f_\Omega^h(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \Omega^h, \qquad \mathcal{L}_{\partial\Omega}^h u^h(\boldsymbol{x}) = f_{\partial\Omega}^h(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in \partial\Omega^h \tag{A.11}$$

and gives an approximation to the continuous problem with a truncation error equal to the order of the finite difference approximation.

In the above notation u^h is a grid function on $\Omega^h \cup \partial\Omega^h$. The discrete operators $\mathcal{L}_\Omega^h$ and $\mathcal{L}_{\partial\Omega}^h$ act on the space of grid functions. Usually, the boundary conditions can be eliminated and (A.11) can be written simply as

$$\mathcal{L}^h u^h = f^h.$$

Example A.3.1.4. Let us consider Poisson's equation (A.8) on domain $\Omega = [0, 1]^2$. At interior grid points, which are not adjacent to the boundary, a second-order central

difference approximation is given by

$$\left(\mathcal{L}_\Omega^h u^h\right)_{i,j} = \frac{-u_{i+1,j} - u_{i-1,j} + 4u_{i,j} - u_{i,j+1} - u_{i,j-1}}{h^2} = \left(f^h\right)_{i,j}$$

We call this discretisation a $5-$point difference operator scheme, as each $\left(\mathcal{L}_\Omega^h u^h\right)_{i,j}$ requires the value of u at 5 different indices. Any points adjacent to the boundary will be replaced by the boundary value at that point dictated by the boundary conditions. For example, if we impose zero Dirichlet BCs then $(u_{\partial\Omega}^h)_{n+1,j} = 0$ and

$$\left(\mathcal{L}_\Omega^h u^h\right)_{n,j} = \frac{-u_{n-1,j} + 4u_{n,j} - u_{n,j+1} - u_{n,j-1}}{h^2} = \left(f_\Omega^h\right)_{n,j}$$

and if we instead impose zero Neumann BCs (with first-order approximation) then

$$\left.\frac{\partial u}{\partial x}\right|_{(n,j)} \approx \Delta_x^+ u_{n,j} = \frac{u_{n+1,j} - u_{n,j}}{h} = 0 \quad \Rightarrow \quad u_{n+1,j} = u_{n,j}$$

and we have

$$\left(\mathcal{L}_{\partial\Omega}^h u^h\right)_{n,j} = \frac{-u_{n-1,j} + 3u_{n,j} - u_{n,j+1} - u_{n,j-1}}{h^2} = \left(f_{\partial\Omega}^h\right)_{n,j}.$$

Similar considerations give $\mathcal{L}_{\partial\Omega}^h u^h$ at other points adjacent to the boundary.

In the next section, we will consider various methods used to solve linear and non-linear discretised equations of the form $\mathcal{L}^h u^h = f^h$ and $\mathcal{N}^h u^h = f^h$ respectively.

A.3.2. Iterative Numerical Methods

Before we introduce any solvers for linear and non-linear systems, it is necessary to briefly review some theory from Linear Algebra. Consider the linear system

$$A\boldsymbol{x} = \boldsymbol{b}$$

where A is an $n \times m$ matrix, $\boldsymbol{x}$ is the $m \times 1$ vector of unknowns and $\boldsymbol{b}$ is an $n \times 1$ vector given by

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,m} \\ a_{2,1} & a_{2,2} & \dots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,m} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}. \tag{A.12}$$

In this thesis, we always suppose that A is a square matrix (i.e. $n = m$). This system has a solution $\boldsymbol{x} = A^{-1}\boldsymbol{b}$, which can be computed directly, if and only if the determinant of A is non-zero, i.e. A is invertible. There are many direct methods for computing this solution each of which has different computational complexity.

Definition A.3.2.1 (Computational Complexity)**.** Consider a linear system of equations in matrix form $A\boldsymbol{x} = \boldsymbol{b}$, where $A \in \mathbb{R}^{n \times n}$ and $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$. If the algorithm used to compute the solution $\boldsymbol{x}$ requires Kn^k binary operations (for arbitrary coefficient K), then we say the algorithm has complexity $\mathcal{O}\left(n^k\right)$ and complexity exponent k.

In Table A.1, we give some common direct solvers and the associated complexities.

Algorithm	Complexity Exponent
Gaussian Elimination [80] (c.200BC)	3.000000
Cholesky Decomposition [22] (1924)	3.000000
LU Factorisation [18] (1938)	3.000000
QR Algorithm [67, 100] (1961)	3.000000
Strassen Algorithm [155] (1969)	2.807355
Coppersmith-Windograd Algorithm [54] (1987)	2.375477
Le Gall Algorithm [103] (2014)	2.3728639

Table A.1: Some algorithms and their associated complexity exponents.

Remark A.3.2.2. Although Gaussian Elimination, Cholesky Decomposition and LU Factorisation have the same complexity exponent, each can prove more efficient in certain circumstances. For example, if we need to solve $A\boldsymbol{x} = \boldsymbol{b}$ for a changing $\boldsymbol{b}$ then we would prefer to use LU factorisation rather than Gaussian Elimination as once we obtain upper and lower triangular matrices L and U such that $A = LU$, the

system $LU\boldsymbol{x} = \boldsymbol{b}$ can be solved with $\mathcal{O}(N^2)$ complexity. We also find that Cholesky decomposition is more efficient than LU factorisation by a factor of 2.

A direct solver will give the exact solution (to machine precision) once the algorithm has been run once; we cannot obtain an approximate solution if we stop the algorithm before the end. However, it is rare that we require the exact solution to $A\boldsymbol{x} = \boldsymbol{b}$ and an approximation to $\boldsymbol{x}$ may be sufficient. In this case, we can consider iterative methods. These take an approximation to $\boldsymbol{x}$ and after each iteration of the method, we aim to obtain a better approximation to $\boldsymbol{x}$. If we repeatedly run an iterative method and the resulting $\boldsymbol{x}$ tends to the result of the direct method we say that the iterative method converges. Crucially, we find that iterative methods can compute an approximation to $\boldsymbol{x}$ far faster than a direct solver computes the exact solution. For a system of non-linear equations, we typically can only use iterative methods to solve them.

A.3.2.1 Linear Iterative Methods

To solve a linear system $A\boldsymbol{x} = b$ we can use a linear iterative method. Examples of some popular linear iterative methods are Jacobi, Gauss-Seidel and SOR [19]. Each of these methods is computationally cheap and easy to implement. As each iteration requires a previous approximation for $\boldsymbol{x}$, the first iteration requires some initial approximation $\boldsymbol{x}^{(0)}$. Suppose we run the iterative method for N iterations, then we can generate a sequence of approximations $\left\{\boldsymbol{x}^{(k)}\right\}_{k=1}^{N}$ which get closer to the true solution $\boldsymbol{x}$ of the linear system via the relation

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c}, \tag{A.13}$$

for each $k = 1, 2, 3, \ldots, N$. In this way the system $A\boldsymbol{x} = b$ is converted into an equivalent system of the form $\boldsymbol{x} = T\boldsymbol{x} + \boldsymbol{c}$ for some fixed matrix T and a vector $\boldsymbol{c}$, neither depending on time sequence k. The Jacobi, Gauss-Seidel and SOR methods all have the form (A.13) with varying T and $\boldsymbol{c}$, as we will now show.

A.3.2.1.1 Jacobi Method

The Jacobi method is one of the simplest iterative methods to implement and forms the basis of other linear iterative methods. Suppose that we have the system $A\boldsymbol{x} = \boldsymbol{b}$ with

A, $\boldsymbol{x}$ and $\boldsymbol{b}$ as in (A.12). Then for square A, with $n = m$, we have

$$b_i = \sum_{j=1}^{n} a_{ij}x_j = a_{ii}x_i + \sum_{j=1, j\neq i}^{n} a_{ij}x_j \tag{A.14}$$

for $i = 1, 2, \ldots, n$ and we rearrange this to

$$x_i = \frac{1}{a_{ii}}\left(\left(-\sum_{j=1, j\neq i}^{n} a_{ij}x_j\right) + b_i\right).$$

Given all the components of $\boldsymbol{x}^{(k-1)}$ (for $k \geq 1$), the components of $\boldsymbol{x}^{(k)}$ are generated by

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(\left(-\sum_{j=1, j\neq i}^{n} a_{ij}x_j^{(k-1)}\right) + b_i\right). \tag{A.15}$$

Let us decompose the square matrix A from (A.12) into a diagonal component D, and the remainder R,

$$D = \begin{bmatrix} a_{1,1} & 0 & \ldots & 0 \\ 0 & a_{2,2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{n,n} \end{bmatrix}, \quad R = \begin{bmatrix} 0 & a_{1,2} & \ldots & a_{1,n} \\ a_{2,1} & 0 & \ldots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \ldots & 0 \end{bmatrix}, \tag{A.16}$$

with

$$A = D + R.$$

In matrix form, we can write the system (A.14), as

$$A\boldsymbol{x} = (D + R)\boldsymbol{x} = \boldsymbol{b} \Leftrightarrow D\boldsymbol{x} = \boldsymbol{b} - R\boldsymbol{x} \Leftrightarrow \boldsymbol{x} = -D^{-1}R\boldsymbol{x} + D^{-1}\boldsymbol{b}.$$

Comparing with the form (A.13), the Jacobi method is

$$\boldsymbol{x}^{(k)} = T_J\boldsymbol{x}^{(k-1)} + \boldsymbol{c}_J \tag{A.17}$$

where $T_J = D^{-1}R$ and $\boldsymbol{c}_J = D^{-1}\boldsymbol{b}$. In practice, this method is implemented using equation (A.15) due to the large amounts of memory required to store the matrices in the matrix form. An advantage of the Jacobi method is that parallel computation can

be implemented, as the computation of $x_i^{(k+1)}$ requires each element in $\boldsymbol{x}^{(k)}$ other than itself. Therefore each component can be computed independently of the others.

Algorithm 2: Algorithm for Jacobi Method: $\boldsymbol{x}^{(k)} \leftarrow JAC(\boldsymbol{x}^{(0)}, A, \boldsymbol{b}, maxit, tol)$.

```
for k = 1 : maxit do
    x^(k) = x^(k-1).
    for i = 1, 2, ..., n do
        x_i^(k) ← (1/a_ii) ( ( - Σ_{j=1, j≠i}^{n} a_ij x_j^(k-1) ) + b_i ).
    end for
    if ||x^(k) - x^(k-1)|| < tol or ||b - A x^(k)|| < tol or k ≥ maxit then
        break;
    end if
end for
```

A.3.2.1.2 Gauss-Seidel Method (GS)

One disadvantage of the Jacobi method is that we must store all components of $\boldsymbol{x}^{(k)}$ and $\boldsymbol{x}^{(k-1)}$ at each iteration. The Gauss-Seidel method uses half the storage of the Jacobi method by updating $x_i^{(k)}$ using the already computed values $\left\{x_j^{(k)}\right\}_{i=1}^{i=j-1}$.

We can write the system $A\boldsymbol{x} = \boldsymbol{b}$ as

$$b_i = \sum_{j=1}^{n} a_{ij}x_j = a_{ii}x_i + \sum_{j=1}^{i-1} a_{ij}x_j + \sum_{j=i+1}^{n} a_{ij}x_j \tag{A.18}$$

for $i = 1, 2, \ldots, n$. Using this, we can write the Gauss-Seidel iterations as

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right) \quad \text{for } i = 1, \ldots, n. \tag{A.19}$$

In matrix form (A.18) can be written as

$$\boldsymbol{b} = D\boldsymbol{x}^{(k)} + L\boldsymbol{x}^{(k)} + U\boldsymbol{x}^{(k-1)}$$

where D is the diagonal matrix (A.16) and L and U are defined by

$$L = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ a_{2,1} & 0 & 0 & \dots & 0 & 0 & 0 \\ a_{3,1} & a_{3,2} & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots \\ a_{n-2,1} & a_{n-2,2} & a_{n-2,3} & & 0 & 0 & 0 \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \dots & a_{n-1,n-2} & 0 & 0 \\ a_{n,1} & a_{n,2} & a_{n,3} & \dots & a_{n,n-2} & a_{n,n-1} & 0 \end{bmatrix}, \tag{A.20}$$

and

$$U = \begin{bmatrix} 0 & a_{1,2} & a_{1,3} & \dots & a_{1,n-2} & a_{1,n-1} & a_{1,n} \\ 0 & 0 & a_{2,3} & \dots & a_{2,n-2} & a_{2,n-1} & a_{2,n} \\ 0 & 0 & 0 & & a_{3,n-1} & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & a_{n-2,n-1} & a_{n-2,n} \\ 0 & 0 & 0 & \dots & 0 & 0 & a_{n-1,n} \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}. \tag{A.21}$$

The system can be written as

$$(D + L)\boldsymbol{x}^{(k)} = -U\boldsymbol{x}^{(k-1)} + \boldsymbol{b}$$

or equivalently

$$\boldsymbol{x}^{(k)} = T_{GS}\boldsymbol{x}^{(k-1)} + \boldsymbol{c}_{GS} \tag{A.22}$$

where $T_{GS} = -(D + L)^{-1}U$ and $\boldsymbol{c}_{GS} = (D + L)^{-1}\boldsymbol{b}$.

Again, the point-wise formula (A.19) is also recommended to save computer memory. One important advantage of the Gauss-Seidel method is that with computer implementation there is no need to allocate two arrays $\boldsymbol{x}^{(k-1)}$ and $\boldsymbol{x}^{(k)}$ in the memory until the updating of $\boldsymbol{x}^{(k)}$ has finished. We can delete every entry $\boldsymbol{x}^{(k-1)}$ as soon as it is no longer needed, which means that only one storage vector is required.

As we will see later, the use of the latest updated values of $\boldsymbol{x}^{(k)}$ leads to a faster convergence rate than the Jacobi method.

Algorithm 3: Algorithm for Gauss-Seidel Method: $\boldsymbol{x}^{(k)} \leftarrow GS(\boldsymbol{x}^{(0)}, A, \boldsymbol{b}, maxit, tol)$.

for $k = 1 : maxit$ **do**
  for $i = 1, 2, \ldots, n$ **do**
    $x_i^{(k)} \leftarrow \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right)$.
  end for
  if $||\boldsymbol{b} - A\boldsymbol{x}^{(k)}|| < tol$ **or** $k \geq maxit$ **then**
    break;
  end if
end for

A.3.2.1.3 Successive Over-Relaxation Method (SOR)

A refinement of the Gauss-Seidel algorithm is Successive Over-Relaxation Method (SOR). The system $\omega A\boldsymbol{x} = \omega \boldsymbol{b}$ can be written

$$\begin{aligned} \omega b_i = \omega \sum_{j=1}^{n} a_{ij} x_j &= \omega a_{ii} x_i + \omega \sum_{j=1}^{i-1} a_{ij} x_j + \omega \sum_{j=i+1}^{n} a_{ij} x_j \\ &= (\omega - 1) a_{ii} x_i + a_{ii} x_i + \omega \sum_{j=1}^{i-1} a_{ij} x_j + \omega \sum_{j=i+1}^{n} a_{ij} x_j \end{aligned} \tag{A.23}$$

and in matrix form this can be written

$$\omega \boldsymbol{b} = (\omega - 1) D\boldsymbol{x}^{(k)} + D\boldsymbol{x}^{(k+1)} + \omega L\boldsymbol{x}^{(k+1)} + \omega U\boldsymbol{x}^{(k)}$$

we rearrange this to

$$(D + \omega L)\, \boldsymbol{x}^{(k+1)} = \omega \boldsymbol{b} - [(\omega - 1)\, D + \omega U]\, \boldsymbol{x}^{(k)}$$

and finally

$$\boldsymbol{x}^{(k+1)} = (D + \omega L)^{-1} \left[\omega \boldsymbol{b} - [(\omega - 1)\, D + \omega U]\, \boldsymbol{x}^{(k)} \right].$$

As we will see in the next section, for appropriate choice of ω we can achieve a convergence rate better than Jacobi and Gauss-Seidel.

A.3.2.1.4 Alternative Linear Iterative Methods

There are many linear iterative solvers which we will not discuss in detail in this thesis. However, for completion, we will mention them here. The Jacobi, Gauss-Seidel and SOR iterative methods discussed so far are all stationary iterative methods, i.e. we can express each of the iterative methods as $\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c}$ where neither T nor $\boldsymbol{c}$ depends on k.

Non-stationary iterative methods exist, these are methods in which the computations involve information which changes at each iteration. Some of the best known non-stationary iterative methods are: Conjugate Gradient [91] (1952), CGNR [91] (1952), CGNE [55] (1955), MINRES [127] (1975), SYMMLQ [127] (1975), BiConjugate Gradient [102, 66] (1976), Chebyshev Iteration [116] (1978), Generalised Minimal Residual [145] (1986), Quasi-Minimal Residual [68] (1991), BiConjugate Gradient Stabilized [160] (1992) and Conjugate Gradient Squared [82] (1997).

In this thesis, we restrict consideration only to the stationary fixed point solvers, and even further, we restrict consideration only to the Jacobi and Gauss-Seidel iterative methods as these are sufficient for our purposes.

A.3.2.2 Convergence of Linear Iterative Methods

All the stationary iterative methods in the previous subsection define a sequence of iterates of the form

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c},$$

for a defined iteration matrix T. We are now interested in the conditions on T which ensure convergence of the iterative method, i.e. $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}$ (the exact solution of $A\boldsymbol{x} = \boldsymbol{b}$). Before introducing the main result of convergence, we present some matrix properties. See [73] for extensive information on the results quoted here.

Definition A.3.2.3 (Symmetric Matrices)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is called a symmetric matrix if it is equal to its transpose A^T .

Definition A.3.2.4 (Diagonally Dominant Matrices)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is said to

be diagonally dominant if it satisfies

$$|a_{ii}| \geq \sum_{j=1, i \neq j}^{n} |a_{ij}| \quad \text{for } 1 \leq i \leq n$$

and is called strictly diagonally dominant if

$$|a_{ii}| > \sum_{j=1, i \neq j}^{n} |a_{ij}| \quad \text{for } 1 \leq i \leq n$$

Definition A.3.2.5 (Positive Definite Matrices)**.** A real symmetric matrix $A \in \mathbb{R}^{n \times n}$ is positive definite if $x^T A x > 0$ for all non-zero vectors $x \in \mathbb{R}^n$. This is equivalent to say that all the eigenvalues of the matrix are strictly positive.

Definition A.3.2.6 (Matrix Convergence)**.** A square matrix $A \in \mathbb{R}^{n \times n}$ is said to be convergent if $\lim_{k \to \infty} A^k = 0$.

If matrix $A \in \mathbb{R}^{n \times n}$ is positive definite or strictly diagonally dominant then A is non-singular and its inverse A^{-1} must exist (all strictly positive eigenvalues). Furthermore, the system $A\boldsymbol{x} = \boldsymbol{b}$ has a unique solution.

Lemma A.3.2.7. For a given matrix $A \in \mathbb{R}^{n \times n}$,

$$||A||_* < 1 \Leftrightarrow \rho(A) := \max_i |\lambda_i| < 1,$$

where λ_i are the eigenvalues of the matrix A and $|| \cdot ||_*$ is any matrix norm. We call $\rho(A)$ the spectral radius of A

Theorem A.3.2.8. If matrix $A \in \mathbb{R}^{n \times n}$ then

$$\lim_{k \to \infty} A^k = 0 \Leftrightarrow \rho(A) < 1.$$

Theorem A.3.2.9 (Spectral Radius and Iterative Convergence)**.** For any $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$, the sequence $\{\boldsymbol{x}^{(k)}\}_{k=1}^{\infty}$ defined by

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c},$$

converges to the unique solution $\boldsymbol{x} = T\boldsymbol{x} + \boldsymbol{c}$ if and only if $\rho(T) < 1$.

We now give a key theorem which explains the widespread use of iterative solvers for linear problems of the form $A\boldsymbol{x} = \boldsymbol{b}$.

Theorem A.3.2.10. Suppose we have the system $A\boldsymbol{x} = \boldsymbol{b}$ with $A \in \mathbb{R}^{n\times n}$ strictly diagonally dominant. The Jacobi and Gauss-Seidel iterative methods all converge to the exact solution $\boldsymbol{x}$ for any $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$. Moreover, the Gauss-Seidel method converges faster than the Jacobi method, i.e. $\rho(T_{GS}) < \rho(T_J) < 1$.

A.3.2.3 Non-Linear Iterative Methods

In general, the equations which result from the variational formulation of imaging models are non-linear, therefore we must now also consider non-linear iterative methods.

In this section, we will discuss three commonly used non-linear iterative solvers, being Newton's method, the Gradient Descent method and Additive Operator Splitting (AOS). We will extensively use the time marching method (a particular case of the Gradient Descent method) and AOS in this thesis.

A.3.2.3.1 Newton's Method

Let $\tilde{\mathcal{N}} : \Omega \subset \mathbb{R}^d \to \mathbb{R}^d$ be a non-linear operator continuously differentiable on $\mathbb{R}^d$. Consider a non-linear system of equations usually represented as

$$\tilde{\mathcal{N}}(\boldsymbol{x}) = \boldsymbol{y},$$

where $\boldsymbol{x} = (x_1, \ldots, x_d)$ is a vector of unknown values and $\boldsymbol{y} = (y_1, \ldots, y_d)$ is a known vector. We can rewrite the equation in the form

$$\mathcal{N}(\boldsymbol{x}) = \tilde{\mathcal{N}}(\boldsymbol{x}) - \boldsymbol{y} = \boldsymbol{0}, \tag{A.24}$$

with $\boldsymbol{0} \in \mathbb{R}^d$ representing the zero vector. We aim to find $\boldsymbol{x} \in \mathbb{R}^d$, a solution of the non-linear equation (A.24). Using the Taylor expansion of $\mathcal{N}$ around $\boldsymbol{x}^{(k-1)}$ we have

$$\mathcal{N}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}^{(k-1)}) + \nabla\mathcal{N}(\boldsymbol{x}^{(k-1)}) \left(\boldsymbol{x} - \boldsymbol{x}^{(k-1)}\right) + \mathcal{O}\left(\left(\boldsymbol{x} - \boldsymbol{x}^{(k-1)}\right)^2\right) = \boldsymbol{0}.$$

and we now use this as a guide to update $\boldsymbol{x}^{(k-1)}$ to $\boldsymbol{x}^{(k)}$ by

$$\mathcal{N}(\boldsymbol{x}^{(k-1)}) = -\nabla\mathcal{N}(\boldsymbol{x}^{(k-1)}) \left(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right).$$

which rearranges to

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \left(\nabla\mathcal{N}\left(\boldsymbol{x}^{(k-1)}\right)\right)^{-1} \mathcal{N}\left(\boldsymbol{x}^{(k-1)}\right)$$

We now use the Jacobian operator to replace $\nabla\mathcal{N}$ in this formulation.

Definition A.3.2.11 (Jacobian Matrix)**.** For a function $\mathcal{N} : \mathbb{R}^d \to \mathbb{R}^d$ we define the Jacobian matrix

$$\mathcal{J} = \begin{bmatrix} \frac{\partial \mathcal{N}_1}{\partial x_1} & \cdots & \frac{\partial \mathcal{N}_1}{\partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathcal{N}_d}{\partial x_1} & \cdots & \frac{\partial \mathcal{N}_d}{\partial x_d} \end{bmatrix}$$

Clearly, if $\mathcal{N} : \mathbb{R}^d \to \mathbb{R}$ then

$$\mathcal{J} = \begin{bmatrix} \frac{\partial \mathcal{N}}{\partial x_1} & \cdots & \frac{\partial \mathcal{N}}{\partial x_d} \end{bmatrix} = (\nabla\mathcal{N})^T.$$

Therefore, assuming $\mathcal{J}$ is invertible, we obtain Newton's method for solving the system of non-linear equations $\mathcal{N}(\boldsymbol{x}) = \boldsymbol{0}$ by applying the following iterative equation

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \mathcal{J}\left(\boldsymbol{x}^{(k-1)}\right)^{-1} \mathcal{N}\left(\boldsymbol{x}^{(k-1)}\right), \quad k \geq 1$$

In practice, computing the inverse of $\mathcal{J}\left(\boldsymbol{x}^{(k-1)}\right)$ may be a difficult task and it is rarely done. Instead, we can look again at the Taylor expansion from earlier and solve the linear system

$$\mathcal{J}\left(\boldsymbol{x}^{(k-1)}\right)\boldsymbol{e}^{(k)} = -\mathcal{N}\left(\boldsymbol{x}^{(k-1)}\right) \quad k \geq 1,$$

where $\boldsymbol{x}^{(0)}$ is given and $\boldsymbol{e}^{(k-1)} = \boldsymbol{x} - \boldsymbol{x}^{(k-1)}$. This assumes that $\boldsymbol{x}$ and $\boldsymbol{x}^{(k-1)}$ are close to one another. We then perform the update

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \boldsymbol{e}^{(k-1)}.$$

Newton's method requires the calculation of the Jacobian at each iteration, and therefore the partial derivatives, which has a high computational cost. If the Jacobian is non-singular at the solution, local quadratic convergence can be proven [92]. So, if we have a good approximation $\boldsymbol{x}^{(0)}$ to $\boldsymbol{x}$, then Newton's method can offer fast convergence [123].

A.3.2.3.2 Gradient Descent Method

Descent methods are a common approach to computing a minimiser of non-linear functionals. Let $\mathcal{F} : \Omega \subset \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable functional which has a minimiser $\boldsymbol{u}$. The key idea of descent methods is updating $\boldsymbol{u}^{(k)}$, the approximation to $\boldsymbol{u}$, by

$$\boldsymbol{u}^{(k)} = \boldsymbol{u}^{(k-1)} - \alpha^{(k-1)}\boldsymbol{s}^{(k-1)}, \quad k \geq 1,$$

requiring an initial approximation $\boldsymbol{u}^{(0)} \in \mathbb{R}^n$. The search direction $-\boldsymbol{s}^{(k-1)}$ is given by some formula along which the new iterate $\boldsymbol{u}^{(k)}$ will be chosen and $\alpha^{(k-1)} \in \mathbb{R}_{>0}$ is a positive step length which can change at each iteration.

The choice of descent direction is crucial to the success of the descent method. For a functional $\mathcal{F}$, the direction of steepest increase is given by $\nabla\mathcal{F}$. Hence, if we set $\boldsymbol{s} = \nabla\mathcal{F}$ we obtain the gradient descent scheme. This is one of the most popular descent methods. Explicitly, the gradient descent scheme is given by

$$\boldsymbol{u}^{(k)} = \boldsymbol{u}^{(k-1)} - \alpha^{(k-1)}\nabla\mathcal{F}\left(\boldsymbol{u}^{(k-1)}\right), \quad k \geq 1.$$

The main characteristic of descent methods is that the iterates decrease the function value

at each stage, i.e.

$$\mathcal{F}\left(\boldsymbol{u}^{(k)}\right) \leq \mathcal{F}\left(\boldsymbol{u}^{(k-1)}\right).$$

when $\alpha^{(k-1)}$ is sufficiently small to satisfy condition. The value of $\alpha^{(k-1)}$ can vary at each iteration or one popular option is to fix it. A gradient descent method with the step length fixed to τ, i.e. $\alpha^{(k-1)} = \tau$ for all $k \geq 1$, is known as the time marching method. The explicit time marching method is given by

$$\boldsymbol{u}^{(k)} = \boldsymbol{u}^{(k-1)} - \tau \nabla \mathcal{F}\left(\boldsymbol{u}^{(k-1)}\right), \quad k \geq 1.$$

These schemes are incredibly easy to implement but have the drawback that they are only stable for small τ. This leads to a large number of iterations being required for convergence to a steady solution. Despite this drawback, many seminal approaches in imaging have employed time marching to obtain a solution, such as Rudin, Osher, and Fatemi [144], Chan and Vese [46] and Chan, Esedoglu and Nikolova [44]. It is possible to reduce the stability restrictions on τ by employing a semi-implicit scheme rather than the explicit scheme, i.e. the gradient is dependent on the current approximation of the solution, so now we consider $\nabla \mathcal{F}\left(\boldsymbol{u}^{(k)}, \boldsymbol{u}^{(k-1)}\right)$.

A.3.2.3.3 Additive Operator Splitting Scheme

Additive Operator Splitting (AOS) was introduced by Gordeziani et al. in 1974 [75] and popularised by Tai [113] in 1991 and Weickert [170] in 1998. It is a semi-implicit scheme designed to solve PDEs involving m-dimensional anisotropic diffusion terms, i.e. PDEs of the form

$$\frac{\partial u(\boldsymbol{x})}{\partial \tau} = \nabla \cdot \left(G\left(u(\boldsymbol{x})\right) \nabla u(\boldsymbol{x})\right) + f(\boldsymbol{x}) \tag{A.25}$$

in $[0, T] \times \Omega \subset \mathbb{R}^{m+1}$, with zero Neumann boundary conditions. The diffusivity function is given by $G\left(u(\boldsymbol{x})\right)$ and the reaction term is $f(\boldsymbol{x})$. We simplify notation by setting $u := u(\boldsymbol{x})$ and $u_\tau = \frac{\partial u}{\partial \tau}$. Equation (A.25) can be written as

$$u_\tau = (G(u)u_{x_1})_{x_1} + \ldots + (G(u)u_{x_m})_{x_m} + f(\boldsymbol{x}), \tag{A.26}$$

in $[0, T] \times \Omega \subset \mathbb{R}^{m+1}$, and with initial condition $u(0, \cdot) = u^{(0)}$. Many variational imaging models result in a PDE of this form, see §2.2, therefore AOS is used extensively in

this thesis. The term u_τ is discretised using a semi-implicit first-order forward finite difference scheme and we use matrix notation to discretise each term in (A.26) as follows

$$\frac{u^{(k+1)} - u^{(k)}}{\tau} = \sum_{\ell=1}^{m} A_\ell \left(u^{(k)}\right) u^{(k+1)} + f$$

where $A_\ell \left(u^{(k)}\right) u^{(k+1)}$ is a discretised version of $\partial_{x_\ell} \left(G \left(u^{(k)}\right) \partial_{x_\ell} u^{(k+1)}\right)$, $\quad k \geq 1$.

Remark A.3.2.12. Weickert also considers the explicit, semi-implicit and fully implicit discretisations of (A.25) in [170]. The authors find that the explicit discretisation gives a tight restriction on the possible time step whereas the semi-implicit and fully implicit schemes are unconditionally stable. The fully implicit scheme is more complex to solve than the semi-implicit scheme and therefore the recommendation is to use the semi-implicit discretisation.

This scheme can be reformulated to

$$u^{(k+1)} - \tau \sum_{\ell=1}^{m} A_\ell \left(u^{(k)}\right) u^{(k+1)} = u^{(k)} + \tau f$$

then

$$u^{(k+1)} = \left(I - \tau \sum_{\ell=1}^{m} A_\ell \left(u^{(k)}\right)\right)^{-1} \left(u^{(k)} + \tau f\right)$$

and finally this can be modified to give the AOS scheme

$$u^{(k+1)} = \frac{1}{m} \sum_{\ell=1}^{m} \left(I - m\tau A_\ell \left(u^{(k)}\right)\right)^{-1} \left(u^{(k)} + \tau f\right). \tag{A.27}$$

Notice that each matrix $A_\ell(u)$ describes diffusion along the x_ℓ axis. Therefore AOS allows us to solve an m-dimensional problem as a series of m 1-dimensional problems and we take the average of these solutions. To be specific, let us consider the 2-dimensional case ($m = 2$), which is of most interest to us in this thesis as most of the images we obtain

are 2-dimensional. In this case, we have

$$\begin{aligned}\left[A_1\left(u^{(k)}\right)u^{(k+1)}\right]_{i,j} &= \left[\partial_{x_1}G\left(u^{(k)}\right)\partial_{x_1}u^{(k+1)}\right]_{i,j}\\ &= \frac{1}{h}\left[G\left(u^{(k)}\right)_{i+1/2,j}\left(\partial_{x_\ell}u^{(k+1)}\right)_{i+1/2,j} - G\left(u^{(k)}\right)_{i-1/2,j}\left(\partial_{x_\ell}u^{(k+1)}\right)_{i-1/2,j}\right]\\ &= \frac{1}{h^2}\left[G\left(u^{(k)}\right)_{i+1/2,j}\left(u_{i+1,j}^{(k+1)} - u_{i,j}^{(k+1)}\right) - G\left(u^{(k)}\right)_{i-1/2,j}\left(u_{i,j}^{(k+1)} - u_{i-1,j}^{(k+1)}\right)\right]\\ &= \frac{1}{h^2}\left[G\left(u^{(k)}\right)_{i+1/2,j}u_{i+1,j}^{(k+1)} + G\left(u^{(k)}\right)_{i-1/2,j}u_{i-1,j}^{(k+1)}\right.\\ &\qquad \left. - \left(G\left(u^{(k)}\right)_{i+1/2,j} + G\left(u^{(k)}\right)_{i-1/2,j}\right)u_{i,j}^{(k+1)}\right]\end{aligned} \tag{A.28}$$

and similarly

$$\begin{aligned}\left[A_2\left(u^{(k)}\right)u^{(k+1)}\right]_{i,j} = \frac{1}{h^2}&\left[G\left(u^{(k)}\right)_{i,j+1/2}u_{i,j+1}^{(k+1)} + G\left(u^{(k)}\right)_{i,j-1/2}u_{i,j-1}^{(k+1)}\right.\\ &\left. - \left(G\left(u^{(k)}\right)_{i,j+1/2} + G\left(u^{(k)}\right)_{i,j-1/2}\right)u_{i,j}^{(k+1)}\right].\end{aligned} \tag{A.29}$$

We see from this that A_1 and A_2 are tridiagonal matrices and therefore $I - m\tau A_\ell(u)$ is also tridiagonal for $\ell = 1, 2$. Therefore, in computing the solution $u^{(k+1)}$, we can utilise the Thomas algorithm [53, 170] which allows us to solve tridiagonal systems of equations in $\mathcal{O}(N)$ time rather than e.g. $\mathcal{O}(N^3)$ for Gaussian elimination (where N is the number of grid points in discretised Ω).

Does the AOS scheme form a discrete scale-space?

As we have introduced the numerical method and its implementation, we now consider some theory for the numerical scheme. Criteria are given which an iterative scheme must satisfy to be a discrete scale-space. In this case we have guarantees about convergence and stability of the method.

For a given image $z(\boldsymbol{x})$, the scale-space representation of $z(\boldsymbol{x})$ is a one-parameter family of smoothed images, parametrised by the size of the smoothing kernel used for suppressing fine-scale structures. A scale-space interpretation of equations of the form (A.25) has been established [96, 168, 169]. This equation creates smoothing scale-spaces whilst also being contrast-enhancing. We desire that the discretisation of the PDE (A.25) also has

these properties. It is found that a discrete scheme of the form

$$u^{(0)} = z, \qquad u^{(k+1)} = Q\left(u^{(k)}\right) u^{(k)}, \qquad \forall k \in \mathbb{N}_0, \tag{A.30}$$

where $Q = (q_{i,j})$, satisfying the criteria (D1)–(D6) below does indeed form a discrete non-linear scale-space. We regard a discrete image as a vector $z \in \mathbb{R}^N$ and denote the index set $\{1, 2, \ldots, N\}$ by J. The criteria which must be satisfied are as follows.

(D1) Continuity in its argument:

$$Q \in C(\mathbb{R}^N, \mathbb{R}^{N \times N})$$

(D2) Symmetry:

$$q_{ij} = q_{ji}, \quad i, j \in J$$

(D3) Unit row sum:

$$\sum_{j \in J} q_{ij} = 1, \quad \forall i \in J$$

(D4) Non-negativity:

$$q_{ij} \geq 0, \quad \forall i, j \in J$$

(D5) Strictly positive diagonal:

$$q_{ii} > 0, \quad \forall i \in J$$

(D6) Irreducibility: Any two elements can be connected by a path with non-vanishing diffusivities. Formally, $\forall i, j \in J$ there exist $k_0, \ldots, k_r \in J$ with $k_0 = i$, and $k_r = j$ such that $q_{k_p k_{p+1}} \neq 0$ for $p = 0, ..., r - 1$.

In the seminal Weickert paper [170], the author proves that once these criteria are all fulfilled, the scheme (A.30) gives a discrete scale-space representation and satisfies the key properties

- **Average grey level invariance.** The average image intensity, given by $\mu_z := \frac{1}{N} \sum_{j \in J} z_j$, is not affected by the discrete diffusion filter.
- **Extremum principle.** We have $\min_{j \in J} z_j \leq u_i^{(k)} \leq \max_{j \in J} z_j$, $\forall i \in J$, $\forall k \in \mathbb{N}_{\geq 0}$.

- **Convergence to a constant steady-state.** We have $\lim_{k\to\infty} u_i^{(k)} = \mu_z,\ \forall i \in J$.

Additionally, there are no restrictions on the time step size as the scheme is unconditionally stable. This demonstrates a clear advantages of AOS over explicit and semi-implicit time marching schemes. This is especially true when $m > 2$.

Remark A.3.2.13. We see that the AOS scheme which we propose solving for our general imaging model, given by (A.27), differs from the required form (A.30) for the discrete scale-space representation. However, we note that we can drop the term τf from consideration without a loss of generality. Therefore, by proving the conditions (D1)–(D6) for (A.27) we do indeed prove that this is a discrete scale-space representation.

Theorem A.3.2.14. The AOS scheme (A.27) with $m = 2$ corresponding to the finite difference equation

$$\begin{aligned}(\mathcal{F}u)^{(k)} := \frac{1}{\tau}u^{(k+1)} &- \frac{1}{2\tau}\left(I - 2\tau A_1\left(u^{(k)}\right)\right)^{-1}\left(u^{(k)} + \tau f\right) \\ &- \frac{1}{2\tau}\left(I - 2\tau A_2\left(u^{(k)}\right)\right)^{-1}\left(u^{(k)} + \tau f\right)\end{aligned}$$

$k = 0, 1, \ldots$, is an $\mathcal{O}\left(\tau + h^2\right)$ approximation [170]. Therefore, from this point of view, the scheme is consistent with the PDE (A.25).

Appendix B

Proof that Condition (I7) Holds in Theorem 3.4.1.3

In this appendix, we provide the proof that condition (I7) of Theorem 3.4.1.3 is satisfied. This is one of the requirements for a unique viscosity solution to exist for the parabolic PDE resulting from minimising the Geodesic model in Chapter 3. This proof also applies to a general class of segmentation models and also to the model in Chapter 4. We restate the condition (I7) below and then prove it.

(I7) For each $R > 0$ there exists a non-decreasing continuous function $\omega_R : [0,\infty) \to [0,\infty)$ satisfying $\omega_R(0) = 0$ such that if $X, Y \in \mathscr{M}^n$ and $\mu_1, \mu_2 \in [0,\infty)$ satisfy

$$\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix} \leq \mu_1 \begin{bmatrix} I & -I \\ -I & I \end{bmatrix} + \mu_2 \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \tag{B.1}$$

then

$$F(t,x,u,p,X) - F(t,y,u,q,-Y) \geq -\omega_R\Big(\mu_1\left(|x-y|^2 + \rho(p,q)^2\right) + \mu_2 + |p-q| + |x-y|\left(\max(|p|,|q|)+1\right)\Big)$$

for all $t \in [0,T], x, y \in \overline{\Omega}, u \in \mathbb{R}$, with $|u| \leq R$, $p, q \in \mathbb{R}^n \backslash \{0\}$ and $\rho(p,q) = \min\left(\frac{|p-q|}{\min(|p|,|q|)}, 1\right)$.

Proof. Using the assumption in (B.1), we write

$$(Xr,r)+(Ys,s)=r^TXr+s^TYs\leq\mu_1\begin{bmatrix}r^T & s^T\end{bmatrix}\begin{bmatrix}I & -I\\ -I & I\end{bmatrix}\begin{bmatrix}r\\ s\end{bmatrix}+\mu_2\begin{bmatrix}r^T & s^T\end{bmatrix}\begin{bmatrix}I & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}r\\ s\end{bmatrix}$$
$$=\mu_1|r-s|^2+\mu_2\left(|r|^2+|s|^2\right).$$

Note that matrix A from (3.21) is a real symmetric matrix and decomposes as $A=QDQ^T=QD^{1/2}D^{1/2}Q^T=BB^T$ with Q orthonormal and $B=QD^{1/2}$. Successively define $r=B(p)e_i$ and $s=B(p)e_i$ for all (e_i), an orthonormal basis, and obtain

$$(Xr,r)=r^TXr=\sum_i(Be_i)^TX(Be_i)=\sum_i e_i^TB^TXBe_i=\text{trace}(B^TXB)=\text{trace}(A(x,p)X).$$

Therefore, we can write

$$\begin{aligned}\text{trace}(A(x,p)X)+\text{trace}(A(y,q)Y)&=(XB(p)e_i,B(p)e_i)+(YB(q)e_i,B(q)e_i)\\&\leq\mu_1|B(p)e_i-B(q)e_i|^2+\mu_2\left(|B(p)e_i|^2+|B(q)e_i|^2\right)\\&=\mu_1\text{trace}\left(\left(B(p)-B(q)\right)^T\left(B(p)-B(q)\right)\right)\\&+\mu_2\left(G(x)+G(y)\right).\end{aligned}$$

We now focus on reformulating the first term, we start by decomposing $A(x,p)$ as follows

$$\begin{aligned}A(x,p)&=\begin{bmatrix}\frac{p_1}{|p|} & -\frac{p_2}{|p|}\\ \frac{p_2}{|p|} & \frac{p_1}{|p|}\end{bmatrix}\begin{bmatrix}0 & 0\\ 0 & G(x)\end{bmatrix}\begin{bmatrix}\frac{p_1}{|p|} & \frac{p_2}{|p|}\\ -\frac{p_2}{|p|} & \frac{p_1}{|p|}\end{bmatrix}\\&=\begin{bmatrix}\frac{p_1}{|p|} & -\frac{p_2}{|p|}\\ \frac{p_2}{|p|} & \frac{p_1}{|p|}\end{bmatrix}\begin{bmatrix}0 & 0\\ 0 & \sqrt{G(x)}\end{bmatrix}\begin{bmatrix}0 & 0\\ 0 & \sqrt{G(x)}\end{bmatrix}\begin{bmatrix}\frac{p_1}{|p|} & \frac{p_2}{|p|}\\ -\frac{p_2}{|p|} & \frac{p_1}{|p|}\end{bmatrix}\end{aligned}$$

so we have $A=BB^T$ where

$$B(p)=\begin{bmatrix}0 & -\frac{p_2}{|p|}\sqrt{G(x)}\\ 0 & \frac{p_1}{|p|}\sqrt{G(x)}\end{bmatrix}.$$

Using this we compute

$$\text{trace}\left(\left(B(p)-B(q)\right)^T\left(B(p)-B(q)\right)\right)=\left|\frac{p}{|p|}\sqrt{G(x)}-\frac{q}{|q|}\sqrt{G(y)}\right|^2.$$

Substituting this in the overall trace sum we have

$$\text{trace}(A(x,p)X) + \text{trace}(A(y,q)Y) \leq \mu_1 \left| \frac{p}{|p|}\sqrt{G(x)} - \frac{q}{|q|}\sqrt{G(y)} \right|^2 + 2\mu_2\theta.$$

as $G(x) < \theta$ (G is bounded) for all $x \in \Omega$. Focussing on the first term in this expression we compute

$$\begin{aligned}
\left| \frac{p}{|p|}\sqrt{G(x)} - \frac{q}{|q|}\sqrt{G(y)} \right|^2 &= \left| \frac{p}{|p|}\sqrt{G(x)} - \frac{p}{|p|}\sqrt{G(y)} + \frac{p}{|p|}\sqrt{G(y)} - \frac{q}{|q|}\sqrt{G(y)} \right|^2 \\
&\leq 2\left(\sqrt{G(x)} - \sqrt{G(y)} \right)^2 + 2G(y) \left| \frac{p}{|p|} - \frac{q}{|q|} \right|^2 \\
&\leq 2\left(\sqrt{G(x)} - \sqrt{G(y)} \right)^2 + 8\theta\rho(p,q)^2
\end{aligned}$$

where $\rho = \min\left(\frac{|p-q|}{\min(|p|,|q|)}, 1 \right)$. This uses inequality $\left| \frac{p}{|p|} - \frac{q}{|q|} \right|^2 \leq 2\rho(p,q)$ (see [76, 77, 78, 81, 104, 134]). We now note that $g(s) = \frac{1}{1+s^2}$ is Lipschitz continuous with Lipschitz constant $\frac{3\sqrt{3}}{8}$.

Note. In the Geodesic Model we fix $G(x) = g(|\nabla z|)$. Therefore, assuming $G(x)$ and $\sqrt{G(x)}$ as Lipschitz requires us to assume that the underlying z is a smooth function [77]. Thankfully, z is typically provided as a smoothed image after some filtering (e.g. Gaussian smoothing) and we can assume regularity of z.

Remark B.0.0.1. It is less clear that $\sqrt{G(x)}$ is Lipschitz, we now prove it explicitly. Firstly, it is relatively easy to prove that

$$\sqrt{G(x)} - \sqrt{G(y)} \leq \frac{2}{3\sqrt{3}} \left| \, |\nabla z(x)| - |\nabla z(y)| \, \right|$$

by letting $K(s) = \sqrt{g(s)}$ and we find $\sup_s |K'(s)| = \frac{2}{3\sqrt{3}}$. We now need to prove that $|\nabla z(x)|$ is Lipschitz also. Take $h(x) = |\nabla z(x)|$, then by a remark in [77], we can conclude $\exists\, \zeta < \infty$ such that

$$\left| \, |\nabla z(x)| - |\nabla z(y)| \, \right| \leq \zeta |x - y|$$

and so $\sqrt{G(x)}$ is Lipschitz with constant $\frac{2}{3\sqrt{3}}\zeta$.

After some computations we obtain

$$\begin{aligned}\left|\frac{p}{|p|}\sqrt{G(x)}-\frac{q}{|q|}\sqrt{G(y)}\right|^2 &\leq 2\left(\frac{2}{3\sqrt{3}}\zeta\right)^2|x-y|^2+8\theta\rho(p,q)^2\\ &=\frac{8}{27}\zeta^2|x-y|^2+8\theta\rho(p,q)^2.\end{aligned}$$

Following the results in [76, 77, 78, 81, 104, 134] we have

$$|\nabla G(x)-\nabla G(y)|\,|p|<\kappa|p||x-y|\leq\kappa\max(|p|,|q|)|x-y|.$$

so overall

$$\langle\nabla G(x),p\rangle-\langle\nabla G(y),q\rangle\leq\kappa\max(|p|,|q|)|x-y|+\eta|p-q|$$

where $|\nabla G(y)|<\eta<\infty$. Finally, we note that

$$-(|p|-|q|)=|q|-|p|\leq\Big||q|-|p|\Big|\leq|p-q|.$$

If we now write

$$\begin{aligned}-(F(t,x,u,p,X)-F(t,y,u,q,-Y))=&\mu\left(\operatorname{trace}(A(x,p)X)+\operatorname{trace}(A(y,q)Y)\right)\\&+\mu\left(\langle\nabla G(x),p\rangle-\langle\nabla G(y),q\rangle\right)\\&-(|p|-|q|)\,k(u)-|p|f(x)+|q|f(y)\\\leq&\,\mu\mu_1\left(\frac{8}{27}\zeta^2|x-y|^2+8\theta\rho(p,q)^2\right)+2\mu\mu_2\theta\\&+\mu\kappa\max(|p|,|q|)|x-y|+\mu\eta|p-q|\\&-(|p|-|q|)\left(k(u)+2\max_{x\in\Omega}f(x)\right)\\\leq&\,\mu\mu_1\left(\frac{8}{27}\zeta^2|x-y|^2+8\theta\rho(p,q)^2\right)+2\mu\mu_2\theta\\&+\mu\kappa\left(\max\left(|p|,|q|\right)+1\right)|x-y|+\mu\eta|p-q|\\&+C_1|p-q|.\end{aligned}$$

where $C_1=\max_{x\in\Omega}\left(k(u)+2\max_{x\in\Omega}f(x)\right)$ (we must assume $k(u),f(x)$ are bounded). Hence

we have

$$F(t,x,u,p,X) - F(t,y,u,q,-Y) \geq$$
$$- \max\left\{\frac{8}{27}\zeta^2\mu, 8\mu\theta, 2\mu\theta, \mu\eta + C_1, \mu\kappa\right\}\left[\mu_1\left(|x-y|^2 + \rho(p,q)^2\right) + \mu_2\right.$$
$$\left. + |p-q| + |x-y|\left(\max(|p|,|q|) + 1\right)\right]$$

and setting $\omega_R = \max\left\{\frac{8}{27}\zeta^2\mu, 8\mu\theta, 2\theta, \eta + C_1, \mu\kappa\right\} R$, this is a non-decreasing continuous function, maps $[0,\infty) \to [0,\infty)$ and $\omega_R(0) = 0$ as required. We have proven that condition (I7) is satisfied. □

Bibliography

[1] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.

[2] R. E. Alcouffe, A. Brandt, J. E. Dendy Jr., and J. W. Painter. The multi-grid method for the diffusion equation with strongly discontinuous coefficients. *SIAM Journal on Scientific and Statistical Computing*, 2(4):430–454, 1981.

[3] H. Ali, N. Badshah, K. Chen, and G. A. Khan. A variational model with hybrid images data fitting energies for segmentation of images with intensity inhomogeneity. *Pattern Recognition*, 51:27–42, 2016.

[4] H. Ali, N. Badshah, K. Chen, G. A. Khan, and N. Zikria. Multiphase segmentation based on new signed pressure force functions and one level set function 2. *Turkish Journal of Electrical Engineering & Computer Sciences*, 25:2943–2955, 2017.

[5] L. Alvarez, P.-L. Lions, and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal on numerical analysis*, 29(3):845–866, 1992.

[6] L. Ambrosio and V. M. Tortorelli. Approximation of functional depending on jumps by elliptic functional via Γ-convergence. *Communications on Pure and Applied Mathematics*, 43(8):999–1036, 1990.

[7] Luigi Ambrosio, Nicola Fusco, and Diego Pallara. *Functions of bounded variation and free discontinuity problems*, volume 254. Clarendon Press Oxford, 2000.

[8] J. F. Aujol, G. Gilboa, T. Chan, and S. Osher. Structure-texture decomposition–modeling, algorithms, and parameter selection. *International Journal of Computer Vision*, 67(1):111–136, 2006.

[9] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[10] N. Badshah and K. Chen. Multigrid Method for the Chan-Vese Model in Variational Segmentation. *Communications in Computational Physics*, 4(2):294–316, 2008.

[11] N. Badshah and K. Chen. On two multigrid algorithms for modeling variational multiphase image segmentation. *IEEE Transactions on Image Processing*, 18(5):1097–1106, 2009.

[12] N. Badshah and K. Chen. Image selective segmentation under geometrical constraints using an active contour approach. *Communications in Computational Physics*, 7(4):759–778, 2010.

[13] N. Badshah, K. Chen, H. Ali, and G. Murtaza. A coefficient of variation based image selective segmentation model using active contours. *East Asian J. Appl. Math.*, 2:150–169, 2012.

[14] E. Bae and X.-C. Tai. *Energy Minimization Methods in Computer Vision and Pattern Recognition, Lecture Notes in Computer Science 5681*, chapter Efficient Global Minimization for the Multiphase Chan-Vese Model of Image Segmentation, pages 28–41. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[15] D. Bai and A. Brandt. Local mesh refinement multilevel techniques. *SIAM Journal on Scientific and Statistical Computing*, 8(2):109–134, 1987.

[16] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[17] X. Bai and G. Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. *International journal of computer vision*, 82(2):113–132, 2009.

[18] T. Banachiewicz. Méthode de résolution numérique des équations linéaires, du calcul des déterminants et des inverses, et de réduction des formes quadratiques. *Bulletin International de l'Académie Polonaise des Sciences et des Lettres. Classe des Sciences. Mathématiques et Naturelles. Série A. Sciences Mathématiques.*, pages 393–404, 1938.

[19] R. Barrett, M. W. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods*, volume 43. Siam, 1994.

[20] A. Beck. *Introduction to nonlinear optimization: theory, algorithms, and applications with MATLAB*, volume 19. Siam, 2014.

[21] A. Benard and M. Gygli. Interactive video object segmentation in the wild. *CoRR*, abs/1801.00269, 2017.

[22] C. Benoit. Note sur une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés à un système d'équations linéaires en nombre inférieur à celui des inconnues.-application de la méthode à la résolution d'un système défini d'équations linéaires (procédé du commandant cholesky). (2):67–77, 1924.

[23] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[24] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[26] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of computation*, 31(138):333–390, 1977.

[27] A. Brandt. Multi-Level Adaptive Solutions to Boundary-Value Problems. *Mathematics of Computation*, 31(138):333–390, 1977.

[28] A. Brandt. Multi-level adaptive techniques for singular-perturbation problems. *NASA-TM-80966, ICASE report 78-18*, 1978.

[29] A. Brandt and O. E. Livne. *Multigrid techniques: 1984 guide with applications to fluid dynamics*, volume 67. SIAM, 2011.

[30] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010.

[31] X. Bresson, S. Esedoḡlu, P. Vandergheynst, J.-P. Thiran, and S. Osher. Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and vision*, 28(2):151–167, 2007.

[32] C. Brito-Loeza and K. Chen. Multigrid algorithm for high order denoising. *SIAM Journal on Imaging Sciences*, 3(3):363–389, 2010.

[33] T. Brox and J. Weickert. Level set segmentation with multiple regions. *IEEE Transactions on Image Processing*, 15(10):3213–3218, 2006.

[34] X. Cai, R. Chan, and T. Zeng. A two-stage image segmentation method using a convex variant of the Mumford–Shah model and thresholding. *SIAM Journal on Imaging Sciences*, 6(1):368–390, 2013.

[35] V. Caselles, F. Catté, T. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numerische mathematik*, 66(1):1–31, 1993.

[36] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic Active Contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[37] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal on Numerical analysis*, 29(1):182–193, 1992.

[38] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20:89–97, 2004.

[39] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011.

[40] A. Chambolle and T. Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[41] T. Chan, S. Esedoḡlu, and M. Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM Journal on Applied Mathematics*, 66(5):1632–1648, 2006.

[42] T. Chan and L. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[43] T. F. Chan, K. Chen, and X.-C. Tai. *Image Processing Based on Partial Differential Equations: Proceedings of the International Conference on PDE-Based Image Processing and Related Inverse Problems, CMA, Oslo, August 8–12, 2005*, chapter Nonlinear Multilevel Schemes for Solving the Total Variation Image Minimization Problem, pages 265–288. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[44] T. F. Chan, S. Esedoḡlu, and M. Nikolova. Algorithms for Finding Global Minimizers of Image Segmentation and Denoising Models. *SIAM Journal on Applied Mathematics*, 66(5):1632–1648, 2006.

[45] T. F. Chan, B. Y. Sandberg, and L. A. Vese. Active contours without edges for vector-valued images. *Journal of Visual Communication and Image Representation*, 11(2):130 – 141, 2000.

[46] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[47] T. F. Chan and L. A. Vese. A level set algorithm for minimizing the Mumford-Shah functional in image processing. In *Variational and Level Set Methods in Computer Vision, 2001. Proceedings. IEEE Workshop on*, pages 161–168. IEEE, 2001.

[48] D. Chen, M. Yang, and L. Cohen. Global minimum for a variant Mumford-Shah model with application to medical image segmentation. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 1(1):48–60, 2013.

[49] K. Chen. *Matrix Preconditioning Techniques and Applications*. Cambridge University Press, 2005.

[50] K. Chen, Y. Dong, and M. Hintermüller. A nonlinear multigrid solver with line Gauss-Seidel-semismooth-Newton smoother for the Fenchel pre-dual in total variation based image restoration. *Inverse Problems and Imaging*, 5(2):323–339, 2011.

[51] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018.

[52] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[53] S. D. Conte and C. W. De Boor. *Elementary Numerical Analysis: An Algorithmic Approach*. McGraw-Hill Higher Education, 1980.

[54] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 1–6. ACM, 1987.

[55] E. J. Craig. The N-step iteration procedures. *Journal of Mathematics and Physics*, 34(1-4):64–73, 1955.

[56] M. G. Crandall, H. Ishii, and P.-L. Lions. User's guide to viscosity solutions of second order partial differential equations. *ArXiv Mathematics e-prints*, June 1992.

[57] D. Cremers, S. Osher, and S. Soatto. Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. *International Journal of Computer Vision*, 69(3):335 – 351, 2006.

[58] B. Dacorogna. *Introduction to the Calculus of Variations*. Imperial College Press, 3rd edition, 2014.

[59] A. Dervieux and F. Thomasset. A finite element method for the simulation of a Rayleigh-Taylor instability. In *Approximation methods for Navier-Stokes problems*, pages 145–158. Springer, 1980.

[60] A. Dervieux and F. Thomasset. Multifluid incompressible flows by a finite element method. In *Seventh International Conference on Numerical Methods in Fluid Dynamics*, pages 158–163. Springer, 1981.

[61] X. Dong, J. Shen, and L. Shao. Submarkov random walk for image segmentation. *IEEE Transactions on Image Processing*, 25(2):516–527, 2016.

[62] J. Duan, W. Lu, Z. Pan, and L. Bai. New second order Mumford-Shah model based on Γ-convergence approximation for image processing. *Infrared Physics and Technology*, 76:641–647, 2016.

[63] Lawrence Craig Evans. *Measure theory and fine properties of functions*. Routledge, 2018.

[64] A. Falcao, J. Udupa, and F. Migazawa. An ultrafast user-steered image segmentation paradigm: live wire on the fly. *IEEE Transactions on Medical Imaging*, 19(1):55–62, 2002.

[65] Herbert Federer. *Geometric measure theory*. Springer, 2014.

[66] R. Fletcher. Conjugate gradient methods for indefinite systems. In *Numerical analysis*, pages 73–89. Springer, 1976.

[67] J. G. F. Francis. The QR transformation a unitary analogue to the LR transformation - part 1. *The Computer Journal*, 4(3):265–271, 1961.

[68] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-hermitian linear systems. *Numerische mathematik*, 60(1):315–339, 1991.

[69] I. M. Gelfand, R. A. Silverman, et al. *Calculus of variations*. Courier Corporation, 2000.

[70] M. Giaquinta and S. Hildebrandt. *Calculus of variations I*, volume 310. Springer Science & Business Media, 2004.

[71] M. Giaquinta and S. Hildebrandt. *Calculus of variations II*, volume 311. Springer Science & Business Media, 2013.

[72] T. Goldstein, X. Bresson, and S. Osher. Geometric applications of the Split Bregman method. *Journal of Scientific Computing*, 45(1-3):272–293, 2010.

[73] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[74] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[75] D.G. Gordeziani and G.V. Meladze. Simulation of the third boundary value problem for multidimensional parabolic equations in an arbitrary domain by one-

dimensional equations. *USSR Computational Mathematics and Mathematical Physics*, 14(1):249 – 253, 1974.

[76] C. Gout and C. Le Guyader. Segmentation of complex geophysical structures with well data. *Computational Geosciences*, 10(4):361–372, 2006.

[77] C. Gout, C. Le Guyader, and L. Vese. Image segmentation under interpolation conditions. *Preprint CAM-IPAM*, page 44 pages, 2003.

[78] C. Gout, C. Le Guyader, and L. A. Vese. Segmentation under geometrical conditions using geodesic active contours and interpolation using level set methods. *Numerical Algorithms*, 39(1-3):155–173, 2005.

[79] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[80] J. F. Grcar. Mathematicians of Gaussian elimination. *Notices of the AMS*, 58(6):782–792, 2011.

[81] L. Guillot and M. Bergounioux. Existence and uniqueness results for the gradient vector flow and geodesic active contours mixed model. *arXiv preprint math/0702255*, 2007.

[82] M. H. Gutknecht. Lanczos-type solvers for nonsymmetric linear systems of equations. *Acta numerica*, 6:271–397, 1997.

[83] Eldad Haber, Lars Ruthotto, Elliot Holtham, and Seong-Hwan Jun. Learning across scales - multiscale methods for convolution neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[84] Wolfgang Hackbusch. *Multi-grid methods and applications*, volume 4. Springer Science & Business Media, 2013.

[85] J. Hadamard. Sur les prolemes aux derives partielles et leur signification physique. *Bulletin of Princeton University*, 13:1–20, 1902.

[86] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[87] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[88] P. W. Hemker. On the order of prolongations and restrictions in multigrid procedures. *Journal of Computational and Applied Mathematics*, 32(3):423 – 429, 1990.

[89] V. E. Henson. Multigrid methods nonlinear problems: an overview. In *Computational Imaging*, volume 5016, pages 36–48, 2003.

[90] Van E Henson. Multigrid methods for nonlinear problems: an overview. Technical report, Lawrence Livermore National Lab., CA (US), 2002.

[91] M. R. Hestenes and E. Stiefel. *Methods of conjugate gradients for solving linear systems*, volume 49. NBS Washington, DC, 1952.

[92] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[93] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms I: Fundamentals*, volume 305. Springer science & business media, 2013.

[94] H. Ishii and M.-H. Sato. Nonlinear oblique derivative problems for singular degenerate parabolic equations on a general domain. *Nonlinear Analysis: Theory, Methods and Applications*, 57(7):1077 – 1098, 2004.

[95] P. Jaccard. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50, 1912.

[96] M. Jeon, M. Alexander, W. Pedrycz, and N. Pizzi. Unsupervised hierarchical image segmentation with level set and additive operator splitting. *Pattern Recognition Letters*, 26(10):1461–1469, 2005.

[97] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[98] M. Klodt, F. Steinbrücker, and D. Cremers. Moment constraints in convex optimization for segmentation and tracking. In *Advanced Topics in Computer Vision*, pages 215–242. Springer, 2013.

[99] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011.

[100] V. N. Kublanovskaya. On some algorithms for the solution of the compute eigenvalue problem. *Zh. Vych, Math.*, 1:555–570, 1961.

[101] D. Y. Kwak. V-cycle multigrid for cell-centered finite differences. *SIAM J. Sci. Comput.*, 21(2):552–564, September 1999.

[102] C. Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[103] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303. ACM, 2014.

[104] C. Le Guyader. *Imagerie Mathématique: segmentation sous contraintes géométriques Théorie et Applications*. PhD thesis, INSA de Rouen, 2004.

[105] C. Le Guyader and C. Gout. Geodesic active contour under geometrical conditions: Theory and 3D applications. *Numerical Algorithms*, 48(1-3):105–133, 2008.

[106] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[107] C. Li, C. Kao, J. Gore, and Z. Ding. Minimization of region-scalable fitting energy for image segmentation. *IEEE Transactions on Image Processing*, 17(10):1940–1949, 2008.

[108] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1925–1934, 2017.

[109] X.-L. Lin, X. Lu, M. K. Ng, and H.-W. Sun. A fast accurate approximation method with multigrid solver for two-dimensional fractional sub-diffusion equation. *Journal of Computational Physics*, 323:204–218, 2016.

[110] C. Liu, M. K.-P. Ng, and T. Zeng. Weighted variational model for selective image segmentation with application to medical images. *Pattern Recognition*, 76:367–379, 2018.

[111] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[112] F. Lu, F. Wu, P. Hu, Z. Peng, and D. Kong. Automatic 3D liver location and segmentation via convolutional neural networks and graph cut. *arXiv preprint arXiv:1605.03012*, 2016.

[113] T. Lu, P. Neittaanmäki, and X-C. Tai. A parallel splitting up method and its application to Navier-Stokes equations. *Applied Mathematics Letters*, 4(2):25 – 29, 1991.

[114] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[115] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE transactions on pattern analysis and machine intelligence*, 17(2):158–175, 1995.

[116] T. A. Manteuffel. Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration. *Numerische Mathematik*, 31(2):183–208, 1978.

[117] C. R. Maurer, R. Qi, and V. Raghavan. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):265–270, 2003.

[118] M. Mohr and R. Wienands. Cell-centred multigrid revisited. *Computing and Visualization in Science*, 7(3-4):129–140, 2004.

[119] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42:577–685, 1989.

[120] A. Napov and Y. Notay. Smoothing factor, order of prolongation and actual multigrid convergence. *Numerische Mathematik*, 118(3):457–483, 2011.

[121] T. Nguyen, J. Cai, J. Zhang, and J. Zheng. Robust interactive image segmentation using convex active contours. *IEEE Transactions on Image Processing*, 21:3734–3743, 2012.

[122] M. Nitzberg, D. Mumford, and T. Shiota. *Filtering, segmentation and depth*. 1993.

[123] J. Ortega and W. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. SIAM, 1970.

[124] S. Osher and R. P. Fedkiw. Level set methods: an overview and some recent results. *Journal of Computational physics*, 169(2):463–502, 2001.

[125] S. Osher and J. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[126] N. Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[127] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM journal on numerical analysis*, 12(4):617–629, 1975.

[128] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1742–1750, 2015.

[129] S. M. Patel and J. N. Dharwa. Accurate detection of abnormal tissues of brain MR images using hybrid clustering and deep learning based classifier methods of image segmentation. *International Journal of Innovative Research in Science, Engineering and Technology*, 6, 2017.

[130] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2017.

[131] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[132] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. An algorithm for minimizing the Mumford-Shah functional. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1133–1140. IEEE, 2009.

[133] A. Protiere and G. Sapiro. Interactive image segmentation via adaptive weighted distances. *IEEE Transactions on image Processing*, 16(4):1046–1057, 2007.

[134] L. Rada. *Variational models and numerical algorithms for selective image segmentation*. PhD thesis, University of Liverpool, 2013.

[135] L. Rada and K. Chen. Improved Selective Segmentation Model Using One Level-Set. *Journal of Algorithms and Computational Technology*, 7(4):509–540, 2013.

[136] Arnold Reusken. *A note on multigrid methods for nonlinear problems*. Eindhoven University of Technology, Department of Mathematics and Computing Science, 1995.

[137] M. Roberts, K. Chen, and K. L. Irion. A convex geodesic selective model for image segmentation. *Journal of Mathematical Imaging and Vision*, Nov 2018.

[138] M. Roberts, K. Chen, and K. L. Irion. Multigrid algorithm based on hybrid smoothers for variational and selective segmentation models. *International Journal of Computer Mathematics*, 0(0):1–25, 2018.

[139] M. Roberts, K. Chen, and K. L. Irion. On an effective multigrid solver for solving a class of variational problems with application to a unified image segmentation model. *Submitted*, 2018.

[140] M. Roberts and J. Spencer. Chan-Vese reformulation for selective image segmentation. *To appear in Journal of Mathematical Imaging and Vision*, 2019.

[141] F. J. Rodrigo, C.and Gaspar and F. J. Lisbona. On a local Fourier analysis for overlapping block smoothers on triangular grids. *Applied Numerical Mathematics*, 105:96–111, 2016.

[142] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[143] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM SIGGRAPH*, 23(3):1–6, 2004.

[144] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[145] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869, 1986.

[146] H. Sagan. *Introduction to the Calculus of Variations*. Courier Corporation, 1969.

[147] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[148] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[149] J. A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[150] J. A. Sethian et al. Level set methods and fast marching methods. *Journal of Computing and Information Technology*, 11(1):1–2, 2003.

[151] J. Shen, Y. Du, W. Wang, and X. Li. Lazy random walks for superpixel segmentation. *IEEE Transactions on Image Processing*, 23(4):1451–1462, 2014.

[152] J. Shen, S. H. Kang, and T. F. Chan. Euler's elastica and curvature-based inpainting. *SIAM Journal on Applied Mathematics*, 63(2):564–592, 2003.

[153] J. Spencer and K. Chen. A convex and selective variational model for image segmentation. *Communications in Mathematical Sciences*, 13(6):1453–1472, 2015.

[154] J. Spencer and K. Chen. Stabilised bias field: Segmentation with intensity inhomogeneity. *Journal of Algorithms and Computational Technology*, 10(4):302–313, 2016.

[155] V. Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

[156] X.-C. Tai, J. Hahn, and G. J. Chung. A fast algorithm for Euler's elastica model using augmented Lagrangian method. *SIAM Journal on Imaging Sciences*, 4(1):313–344, 2011.

[157] A. N. Tikhonov. On the solution of ill-posed problems and the method of regularization. In *Doklady Akademii Nauk*, volume 151, pages 501–504. Russian Academy of Sciences, 1963.

[158] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[159] A. Tsai, A. Yezzi, and A. S. Willsky. Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. *IEEE Transactions on Image Processing*, 10(8):1169–1186, 2001.

[160] H. A. Van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.

[161] S. P. Vanka. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *Journal of Computational Physics*, 65(1):138–158, 1986.

[162] L. A. Vese and T. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *International journal of computer vision*, 50(3):271–293, 2002.

[163] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Analysis Machine Intell.*, 13(6):583–598, 1991.

[164] W. L. Wan and T. F. Chan. Robust multigrid methods for nonsmooth coefficient elliptic linear systems. *Journal of Computational and Applied Mathematics*, 123:323–352, 2000.

[165] G. T. Wang, W. Q. Li, S. Ourselin, and T. Vercauteren. Automatic brain tumor segmentation using cascaded anisotropic convolutional neural networks. *https://arxiv.org/pdf/1709.00382.pdf*, preprint, 2017.

[166] G. T. Wang, W. Q. Li, M. A. Zuluaga, R. Pratt, P. A. Patel, M. Aertsen, T. Doel, A. L. David, J. Deprest, S. Ourselin, and T. Vercauteren. Interactive medical image segmentation using deep learning with image-specific fine-tuning. *CoRR*, abs/1710.04043, 2017.

[167] X. F. Wang, D. S. Huang, and H. Xu. An efficient local Chan-Vese model for image segmentation. *Pattern Recognition*, 43(3):603–618, 2010.

[168] J. Weickert. Theoretical foundations of anisotropic diffusion in image processing. In *Theoretical foundations of computer vision*, pages 221–236. Springer, 1996.

[169] J. Weickert. *Anisotropic Diffusion in Image Processing*. B.G. Teubner, 1998.

[170] J. Weickert, B. M. Ter Haar Romeny, and M. A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7(3):398–410, 1998.

[171] P Wesseling. Cell-centered multigrid for interface problems. *Multigrid Methods: Theory, applications, and supercomputing*, 110:631–641, 1988.

[172] P. Wesseling. Introduction to multigrid methods. Technical report, Institute For Computer Applications in Science and Engineering Hampton VA, 1995.

[173] P. Wesseling. *An Introduction to Multigrid Methods*. An Introduction to Multigrid Methods. R.T. Edwards, 2004.

[174] N. Xu, B. Price, S. Cohen, J. Yang, and T. S. Huang. Deep interactive object selection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[175] J. Yuan, E. Bae, X.-C. Tai, and Y. Boykov. A spatially continuous max-flow and min-cut framework for binary labeling problems. *Numerische Mathematik*, 126(3):559–587, 2013.

[176] J. P. Zhang, K. Chen, and B. Yu. A 3D multi-grid algorithm for the CV model of variational image segmentation. *International Journal of Computer Mathematics*, 89(2):160–189, 2012.

[177] K. Zhang, L. Zhang, H. Song, and W. Zhou. Active contours with selective local or global segmentation: a new formulation and level set method. *Image and Vision Computing*, 28(4):668–676, 2010.

[178] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. *Computer Vision and Image Understanding*, 80(3):295–314, 2000.

[179] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[180] W. Zhu, X.-C. Tai, and T. F. Chan. Image segmentation using Euler's elastica as the regularization. *Journal of Scientific Computing*, 57:414–438, 2013.

[181] Y. Zhu. Analysis of a multigrid preconditioner for Crouzeix-Raviart discretization of elliptic partial differential equation with jump coefficients. *Numerical Linear Algebra with Applications*, 21(1):24–38, 2014.

[182] William P Ziemer. *Weakly differentiable functions: Sobolev spaces and functions of bounded variation*, volume 120. Springer Science & Business Media, 2012.